Oracle® Enterprise Communications Broker

User's Guide

Release P-CZ2.2.0

June 2018

Oracle Enterprise Communications Broker User's Guide, Release P-CZ2.2.0

Contents

3 Registrar and Authentication

4 Getting Started

15 Troubleshooting and Maintenance

About This Guide

The *Oracle® Enterprise Communications Broker User Guide* provides the following information about the Oracle Enterprise Communications Broker (ECB) hardware and software.

- Configuration examples
- Configuring SIP signaling management
- Configuring dial plans, agents, users, policies, registrars, LDAP, ECB sync, Header Manipulation Rules, and routing
- Maintenance and troubleshooting

Related Documentation

The following table describes the documentation set for the ECB.

Document Name	Document Description
Administrator's Guide	Describes how to deploy the system.
Embedded Help system	Contains task-oriented topics for configuring, administering, maintaining, and troubleshooting the ECB hardware and software.
Release Notes	Contains information about the current release, including specifications, requirements, new features, enhancements, inherited features, known issues, caveats, and limitations.
SBC Family Security Guide	Provides information about security considerations and best practices from a network and application security perspective for the Enterprise family of products.
User's Guide	Describes how to configure SIP signaling management and how to tailor the system to specific needs.

Revision History

The following table describes updates to this guide.

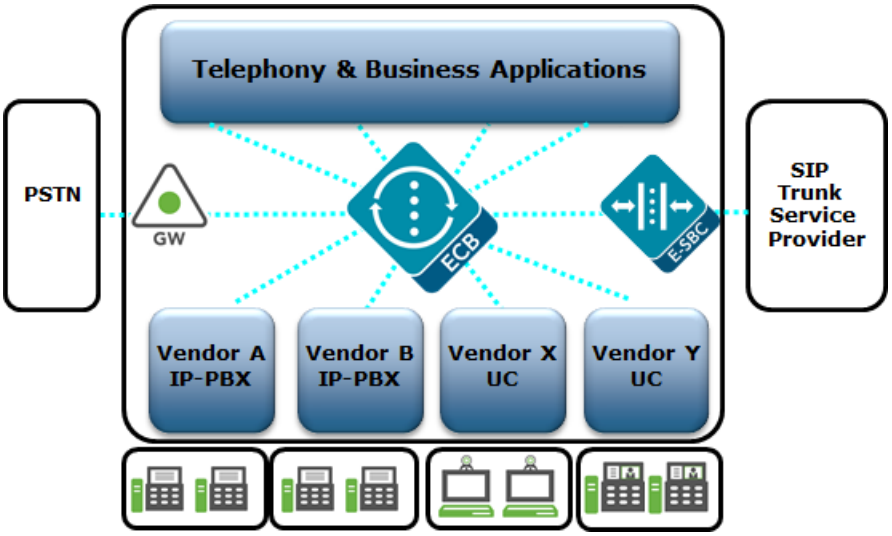
Date	Description
July 2017	• Initial Release
June 2018	• Edits the field label names in steps 10 and 11 in the "Configure LDAP Server Access Fields" topic to match the GUI.

1

Oracle Enterprise Communications Broker (ECB) Overview

The Oracle Enterprise Communications Broker is an enterprise-class, core signaling component designed to simplify communications networks. It combines innovative approaches toward dial plan management and SIP topology-aware routing with a purpose-built, intuitive GUI interface. While at its best in signaling environments comprised of products and solutions from multiple vendors, it is useful for consolidating policy enforcement decisions, integrating third-party applications, and managing a network-wide routing topology even in homogenous architectures.

The Oracle Enterprise Communications Broker is typically deployed in the core of a multi-vendor communications network where multiple UC, PBX and service provider trunk interfaces must be interconnected. It normalizes communications between disparate premise-based systems and connects them to service provider networks and hosted applications through E-SBCs.



Key benefits include:

- Increases scalability and simplicity
- Protects and extends investments in legacy communications infrastructure
- Reduces operations expenses
- Improves network availability
- Services and Applications

Oracle Enterprise Communications Broker operational functionality focuses around the following:

- SIP Signaling Management—The functional components of the Oracle Enterprise Communications Broker's software architecture for SIP signaling management focus around its dial plan, and its routing engine. These two components represent the foundation

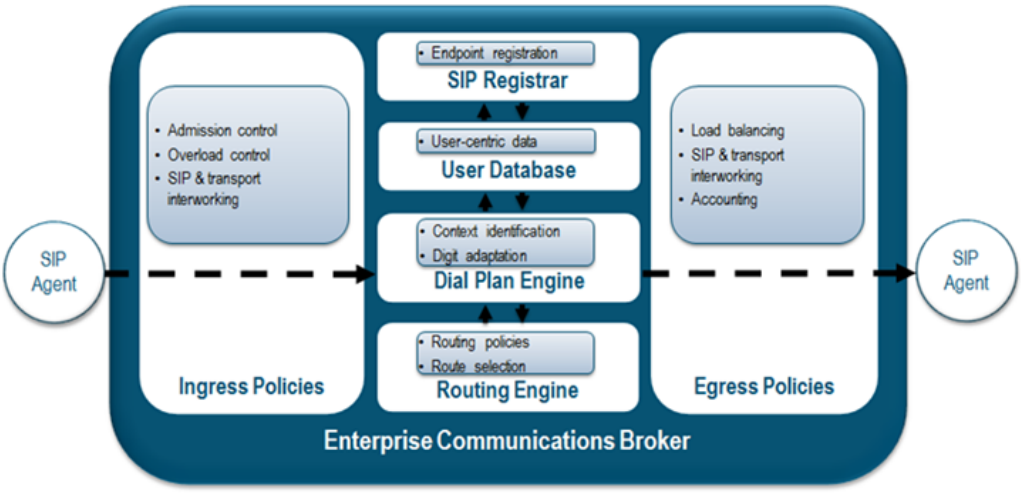
of the Oracle Enterprise Communications Broker's core SIP processing engine, and were specifically crafted to address, in a generic sense, the problems arising from the organic evolution of SIP-based enterprise communications networks.

- SIP Registrar—Provides a centrally-deployed location service for the enterprise.
- User Authentication—Provides for operation with an internal or external authentication resource, such as Active Directory, for authorization and authentication of users registering at the Oracle Enterprise Communications Broker.
- Header Manipulation—Provides telephony engineering with a means of assembling signaling header information specifically for the enterprise's operations, conformance and interoperability needs.

This document provides operational explanations and configuration instructions for each of these.

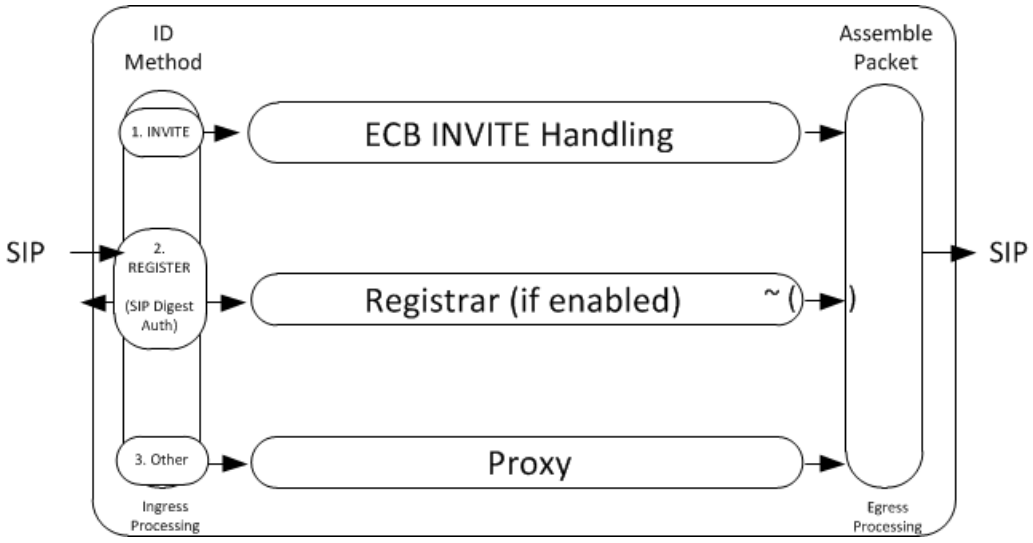
Packet Processing by the Oracle Enterprise Communications Broker

The following sections describe, at a high level, the processing performed by the elements of the Oracle Enterprise Communications Broker to all traffic that it handles. Understanding this processing provides insight into configuration and troubleshooting tasks. Individual elements are documented in deeper detail in ensuing chapters. The diagram below provides visual context for these elements' interactions.

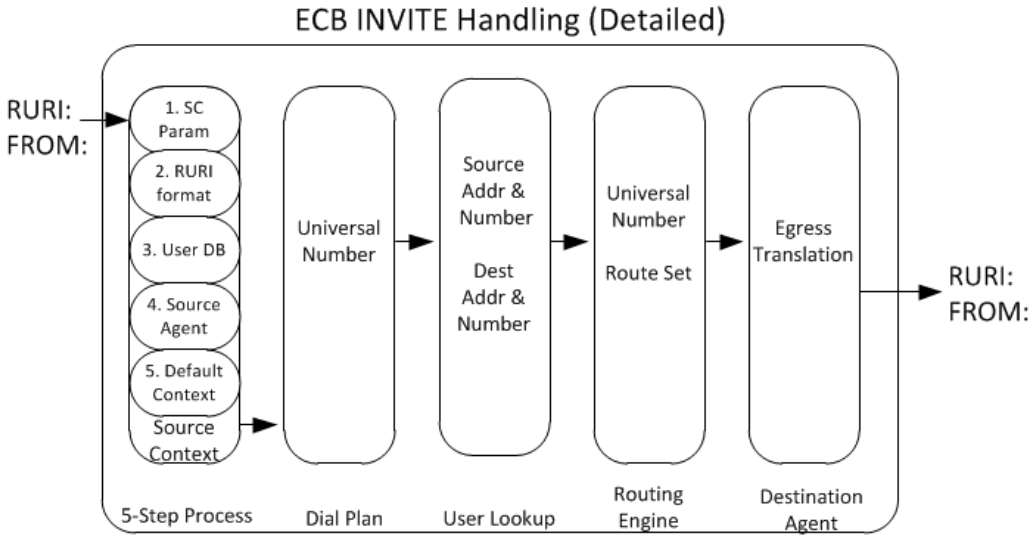


The image below displays Oracle Enterprise Communications Broker processing of different signaling messages, including:

- INVITE—Pass through the INVITE handling processes, which includes number normalization, route optimization and multi-contact support.
- REGISTER—When configured as a SIP Registrar, registration traffic passes into the registrar for authorization, authentication and caching. There are multiple means of performing authorization and authentication.
- Other—All other signaling traffic is proxied, based on RFC 3261 standards, including the insertion of VIA and Route Record headers to keep the Oracle Enterprise Communications Broker in the path of each applicable dialog's traffic.

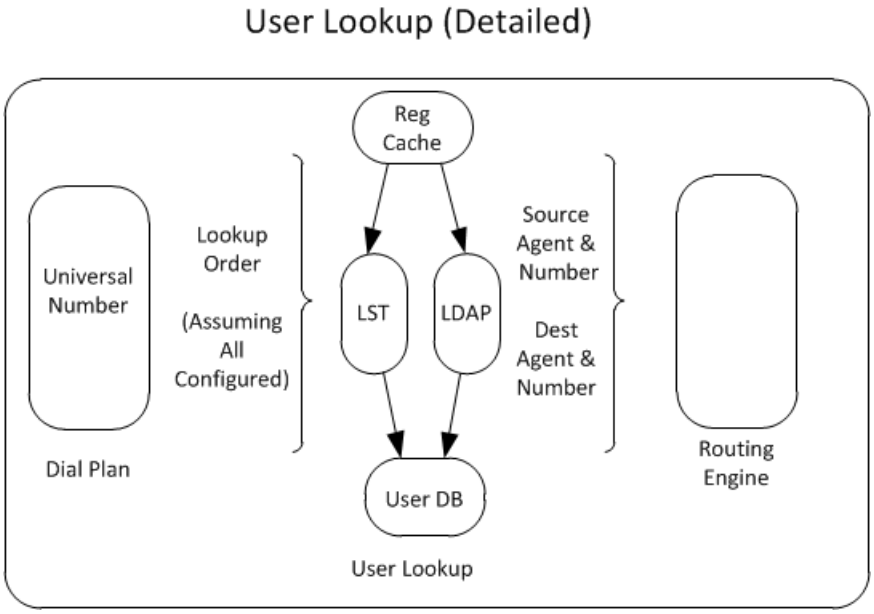


The next diagram displays the key processing elements handling an INVITE, including number normalization, based on context, end station lookup and recursive route set creation.



The next diagram details the elements the system examines to perform user lookup. The Oracle Enterprise Communications Broker queries each of the objects shown in the diagram to identify the destination agent. Having identified the applicable agent, the user lookup hands everything the routing engine needs to recursively specify hop-by-hop routing through agents to reach the target.

Note that utilization of LST versus LDAP resources are independent and exclusive of each other. Either the LST or the LDAP resources perform the functions needed after registration cache procedures. The Oracle Enterprise Communications Broker allows you to configure either LST or LDAP resources.



The following subsections explain this INVITE handling detail.

Ingress INVITE Processing

When an packet arrives at a Oracle Enterprise Communications Broker ingress interface, standard link and network layer processing occurs to prepare the data for processing within the device. Subsequently, the Oracle Enterprise Communications Broker performs admission and overload control procedures to ensure it is both appropriate and possible to proceed with further processing. As discussed, ensuing processing is based on traffic type, of which INVITE processing is key to the overall purpose of the Oracle Enterprise Communications Broker.The sections below describe further INVITE processing.

Identifying Source Context

When receiving an inbound SIP message, the Oracle Enterprise Communications Broker first determines the *source context* of the calling party. This allows the Oracle Enterprise Communications Broker to interpret the dialed digits appropriately.

For example, a user dialing 911 in the United States has different expectations than a user dialing extension 911 in a European office.

The system performs four steps sequentially to identify the source context. If a step identifies a source context, the system skips the next steps and provides the information to the dial plan engine for subsequent processing. These steps include:

1. The system searches the FROM address in the signaling for a source context (SC) parameter. This parameter, if present, identifies the UA's source context.
2. If the number presented in the RURI begins with a "+" sign, assume the RURI is an e.164 number and bypass the source context identification.
3. The system treats the digits received in the userinfo portion of the From header as a universal address and checks to see if the calling party is in its User database.
    a. If there is a match and the user has a source context configured, the system uses that as the call's source context.

   b. If the user has no source context configured, the system check the user's home agent for a source context and, if configured, uses that as the call's source context.

4. The system looks for a Source Context value in the configuration for the Agent from which the message was received.
5. If the above fail, the system uses the default Source Context, as configured in the SIP Interface settings.

Dial Plan Processing

The dial plan receives the dialed digits and the source context of that signaling message, and uses the rules associated with the identified context to prepare the *universal address* from the digits that were dialed. As described in the section on the dial plan engine, this may involve stripping routing digits out of the dial sequence, adding addressing digits into the sequence, or both.

The result of the dial plan processing yields the universal address that the system passes into the routing engine.

Route Engine Processing

The route engine receives the information from the dial plan lookup and builds a search key based on the calling number, called number, source agent, and destination agent for that call. As described in the section on Oracle Enterprise Communications Broker Routing, it recursively processes each route lookup result to construct full route sets.

Egress Processing

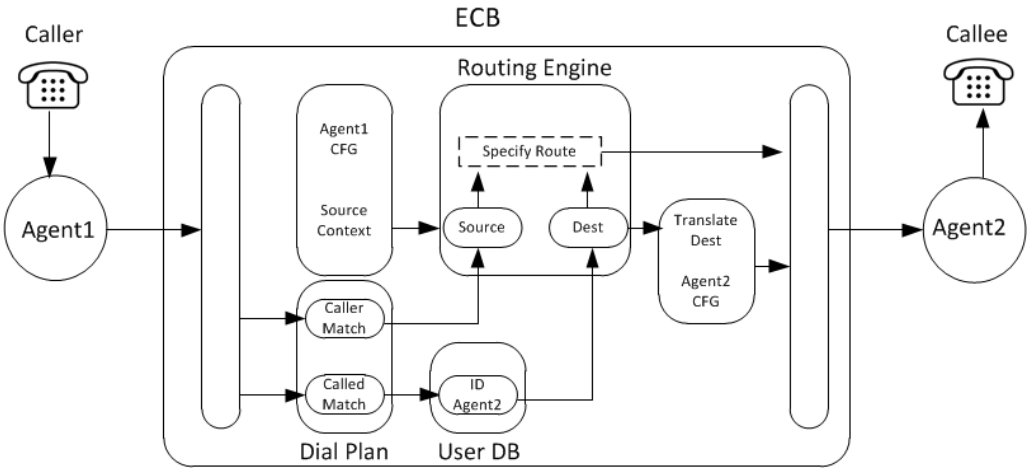
Now that the Oracle Enterprise Communications Broker has a fully qualified universal address, a route or set of routes to use for processing that call, it will prepare the universal address to suit the formatting requirements of the destination. It does this by looking for the Number Translation Mode of the *destination agent* (not any intermediate agents) and applying the transformation identified within that agent's configuration.

Lastly, the message is sent on its way based on the most preferred route. If that route fails, the Oracle Enterprise Communications Broker will try all subsequent route sets that it learned via the routing engine, in order from least cost to highest cost. This may also involve re-writing the universal address to suit the new "last hop".

Call Handling Example

This section provides an illustration of call handling through the Oracle Enterprise Communications Broker. This example assumes no registration cache and no LDAP configuration.

Consider the diagram below, showing a simple, intra-organization call passing through the Oracle Enterprise Communications Broker.



A user within the organization calls another within the organization residing on a different PBX. The call proceeds through the components of the Oracle Enterprise Communications Broker using the following steps:

1. Call Received by Ingress Processing.
2. Ingress processing hands FROM and Request-URI to Dial Plan.
3. System runs 5-step process to ID source context.
4. Dial Plan normalizes Source and Destination Numbers.
5. FROM handed to Routing Engine as Source.
6. Dial Plan hands Request-URI to User DB to ID Home Agent.
7. User DB hands Request-URI to Routing Engine.
8. Routing Engine uses FROM and Agent1 CFG to create new Source.
9. Routing Engine builds Route.
10. Routing Engine hands Request-URI to Agent 2 configuration.
11. Agent 2 configuration translates Request-URI into format compatible with Agent 2.
12. Agent 2 configuration hands Request-URI to Egress processing.
13. Egress processing builds INVITE.
14. Egress processing sends new INVITE to Agent2.

2
SIP Signaling Management

Oracle Enterprise Communications Broker SIP signaling management requires review of the following topics:

- Dial Plan—Normalizes dialing numbers.
- Contexts—Provides rules for dialing number normalization.
- Agents—Establish hop locations for routes.
- Routing—Builds hop-by-hop path to the end-station's target agent.

This section provides explanations of the elements and their operation within the Oracle Enterprise Communications Broker.

The Oracle Enterprise Communications Broker Dial Plan

The Oracle Enterprise Communications Broker's dial plan engine was designed from the ground up to simplify the administration of common, real-world dialing behaviors. Conceptually, the dial plan engine allows administrators to define the rules by which dialed digit strings are built up, or broken down into "universal addresses". A universal address may be thought of as an E.164 number, although this is not strictly required. Universal numbers are required to be globally unique, not E.164-compliant.

These rules are then grouped into a foundation data structure in the Oracle Enterprise Communications Broker, the *context*. The concept of a context is fundamental to the operation of the Oracle Enterprise Communications Broker's dial plan configuration, and is discussed below.

The dial plan engine serves two purposes. First, it constructs universal addresses from input received. Second, it prepares egress translation from universal addresses into contextually-appropriate addresses based upon a message's destination. An example of the latter is the system creating a URI for a remote phone that needs to be addressed with four digits rather than a fully-qualified E.164 number.

Oracle Enterprise Communications Broker Contexts

Simply put, a context is a collection of rules that serve to manipulate strings of dialed digits. It is important to note that in most real-world use cases, contexts are associated with a PBX or branch office. That is, users associated with a given PBX are all subject to its rules for making telephone calls, such as:

- Each user on the PBX dials the same digit for seizing an outside line;
- All users may be able to reach other extensions on that PBX by dialing short dial strings;
- All users in that environment have access to the same 'tie lines'.

The rules that govern how to interpret the series of digits do not differ from user to user within that PBX.

Note that this is *not taking user-based entitlements into consideration.* For example, users within the same context all dial the same videoconferencing terminal in the corporate boardroom using the same series of digits, even though not all of the users are authorized to use that equipment.

Determination of a SIP message's "source context" is critically important. This is covered in more detail in the "Ingress Processing" section below. Phone numbers within a SIP message may have vastly different interpretations when, for example, a user dials "0" from two different branch offices within the same enterprise. The *context* of the dialing user differentiates the dialed pattern for the Oracle Enterprise Communications Broker.

Terminology used to define contexts applicable to the Oracle Enterprise Communications Broker is presented in the table below. Ensuing sections go into deeper detail on these context types, as needed.

Context Type	Definition
Geographic context	This type of context is a collection of rules that define the dialing patterns applicable to that geography, usually a country. These rules are outside of an enterprise's control and are pre-configured for you on the Oracle Enterprise Communications Broker by Oracle. You can, however, extend or modify these rules, if necessary.
Corporate context	Rules defined by the enterprise that specify routing, policy, access code and extension range dialing patterns. **Note:** Rules may vary based on applicable PBX or branch office. Context hierarchy manages these variations.
Source Context	The context used when an Agent provides context detail for a given call. This is also the context within which a given user resides via configuration, to be understood as a user's default location.
Source Context Param	"sc", meaning source context, is the syntax for a parameter on the FROM heaer presented by equipment external to the Oracle Enterprise Communications Broker that specifies the context from which the call originated. When presented, this context's rules are always applied.

Refer to the chapters on Dial Plan configuration for instructions on the related fields.

Context Hierarchy

Contexts within the Oracle Enterprise Communications Broker may be defined *hierarchically*, to offer a parent/child inheritance relationship. This is done to avoid data duplication and redundant configuration.

For example, a large enterprise may have a corporate dial plan (common phone numbers for the IT help desk, employee benefits group, travel desk, etc.) that is consistent among all branch

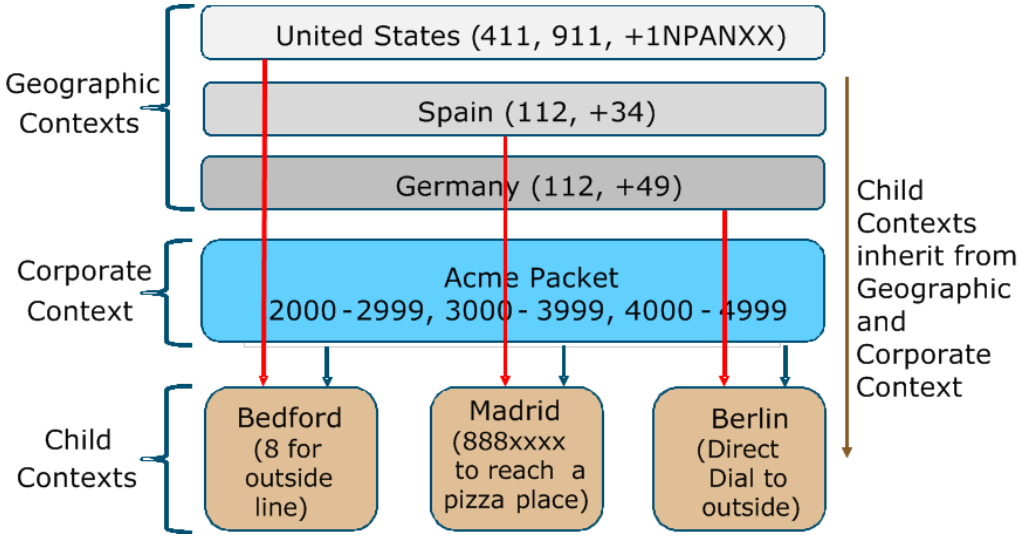
offices, and unique extension ranges per branch location. By defining common data in a "parent" context, each child context will inherit these common dial plan values and avoid the need for configuring each of them over and over for each branch office turn-up.

Each dialing context may have one corporate parent (for inheriting dialing rules that are unique for that enterprise) and one geographic parent (for inheriting common dialing rules pertaining to that branch office's physical location). Geographic and corporate contexts are described in the following sections.

Geographic Contexts

A geographic context is the set of rules for dialing within a given geography. It does not matter if you live in New York City or in Los Angeles, you'll still dial 011 for an international long distance call and 911 for emergency services because those are both part of the dial plan for the United States (or, more technically, the North American Numbering Plan or NANP). The Oracle Enterprise Communications Broker ships with geographic dial plans for the fifty most populous countries on Earth. This default data may be overridden by Oracle Enterprise Communications Broker administrators, or refreshed with future data (to account for changes in the ITU dial plans, for example).

Each context that an administrator defines on the Oracle Enterprise Communications Broker may have a geographic parent, configured as a geographic location. By configuring a geographic location, that child context inherits the dialing patterns for that geography. There is no need to configure the child contexts with rules for 011+, 911, 411 and so forth. They inherit these rules because they participate in that geography's parentage relationship.

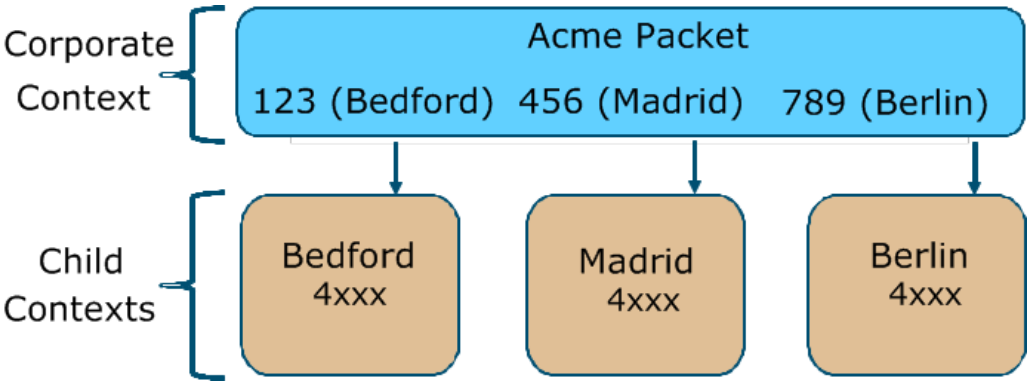


Note:

The digit ranges within the child contexts do not overlap, presenting a simple means of identifying context. This represents a Small Enterprise Oracle Enterprise Communications Broker Configuration Model. The corporate dialing patterns are configured on the corporate context once. When a caller dials 3xxx from any child context, the system always sends the call to Madrid.

Corporate Contexts

As opposed to geographic contexts, which are common for all telephone calls throughout the world and therefore may be supplied with the Oracle Enterprise Communications Broker software, corporate contexts are company-specific and define the dialing rules for the enterprise. This may include all branch offices/remote offices/PBXs and so forth.



Note:

In contrast to the example shown previously, the digit ranges at the child contexts overlap. This represents a Large Enterprise Oracle Enterprise Communications Broker Configuration Model. In this case, the system uses the dialed prefix to identify the child context. Each child context inherits the dial patterns of the parent to know where to send a call. Each child knows to send calls with the prefix 123 to a Bedford tie line, 456 to Madrid and 789 to Berlin. Each pattern need only be configured once, on the parent (Acme Packet) context.

Oracle Enterprise Communications Broker Agents

An agent defines a signaling endpoint. It is a next hop signaling entity that applies traffic shaping attributes to flows. Agents provide important properties for Oracle Enterprise Communications Broker operation, including:

- Transit and termination points for Oracle Enterprise Communications Broker routes; and
- Context identification for use by the Oracle Enterprise Communications Broker dial plan.

Agents can include the following types of devices:

- Softswitches
- SIP proxies
- Application servers
- SIP gateways
- Indirect Agents

For each agent, concurrent session capacity and rate attributes can be defined. The Oracle Enterprise Communications Broker can provide load balancing across the defined agents.

Why You Need Agents

You can use agents to define hops the Oracle Enterprise Communications Broker can use in a signaling path. You can also use them to define and identify preferred carriers. This set of carriers is matched against the local policy for requests coming from the agent. You can also set traffic constraints against specific hops via agent configuration.

In addition to functioning as a logical next hop for a signaling message, agents can provide information regarding next hops or previous hops for SIP packets, including providing a list of equivalent next hops.

How to Use Agents

Consider agents as next-hops within routing paths. Before configuring an agent, map out your session network and identify all potential agents. Each agent should be seen as a best hop based on its location, adjacencies and path costs. Redundant paths are also configurable using agents, allowing manual cost configurations for what may otherwise be equal cost paths.

In addition, consider the users for which each agent is a first hop. Agent configuration provides a method of defining routing and policy configuration for groups of users. Agents also provide a mechanism for defining source context for groups of users.

In some cases, specific addressing is not available or needed to access signaling endpoints. It may be that routing to a target domain is preferable to routing to a specific agent. In these cases, you can configure an agent using, for example, only the target domain name rather than a specific endpoint. When doing this, you assume that the domain itself is able to route to any further hops needed to reach the UA and that the same policies must be utilized from all traffic from that domain.

Agent Groups

Agent groups contain multiple agents. Members of an agent group are logically equivalent (although they might vary in their individual constraints) and can be used interchangeably as transit targets for SIP traffic. For one reason or another, a given agent may not be able to service traffic. Users configure agent groups to establish multiple transit destinations for purposes such as redundancy.

Examples of agent groups include the following:

- Application Server cluster
- Media Gateway cluster
- Softswitch redundant pair
- SIP Proxy redundant pair
- Gatekeeper redundant pair

Agent group members do not need to reside in the same domain, network, or realm. The Oracle Enterprise Communications Broker can allocate traffic among member agents regardless of their location. It uses the allocation strategies configured for an agent group to allocate traffic across the group members.

The user configures agent groups from the GUI's Agent configuration dialog. This configuration consists of simply naming the group, selecting the allocation strategy, selecting recursion preference and adding the agent group members.

Having configured the group, the user then configures agent group names as:

- A `Dest agent` in a routing table entry
- A `Route` in a routing table entry
- A user's `Home agent` in the user database
- A `Default home agent` within an LDAP query

The syntax for these entries appears as the word 'group' followed by a colon (:) and the group name.

```
group:MyGroupName
```

When configuring the group, the user selects between the following allocation strategies to define the method of selecting the next member of the group for a connection attempt if the previous connection attempt fails:

Allocation Strategy	Description
Hunt	Oracle Enterprise Communications Broker selects the agents in the order in which they are configured in the agent group. If the first agent is in service, and has not exceeded any defined constraints, all traffic is sent to the first agent. If the first agent is out of service, or is in violation of constraints, all traffic is sent to the second agent. And so on for all agents in the agent group. When the first agent returns to service, the new traffic is routed back to it.
Round robin	Oracle Enterprise Communications Broker selects each agent in the order in which it is configured, routing a session to each agent in turn.

To summarize, agent group operation requires the following configuration:

- Two or more agents
- An agent group containing those agents
- A route, user configuration or LDAP query that directs traffic to that group

Recursion

Agent groups use a recursive process to communicate with agent group members. This recursion behavior is specified by the allocation strategy. The user can optionally configure the Oracle Enterprise Communications Broker to attempt communications with only a single member of the agent group by leaving the **Try All** control unchecked (disabled).

The agent group performs its agent selection rotation process independently of this recursion setting. Each allocation strategy rotates agent selection as a means of selecting the first agent to try. This ensures that the system continues to use each agent in the group as a message target.

Routing paths may traverse multiple agent groups. The system, however, only performs recursion on the last agent group in the path. This reduces what could otherwise become an inordinate number of connection attempts.

Oracle Enterprise Communications Broker Routing

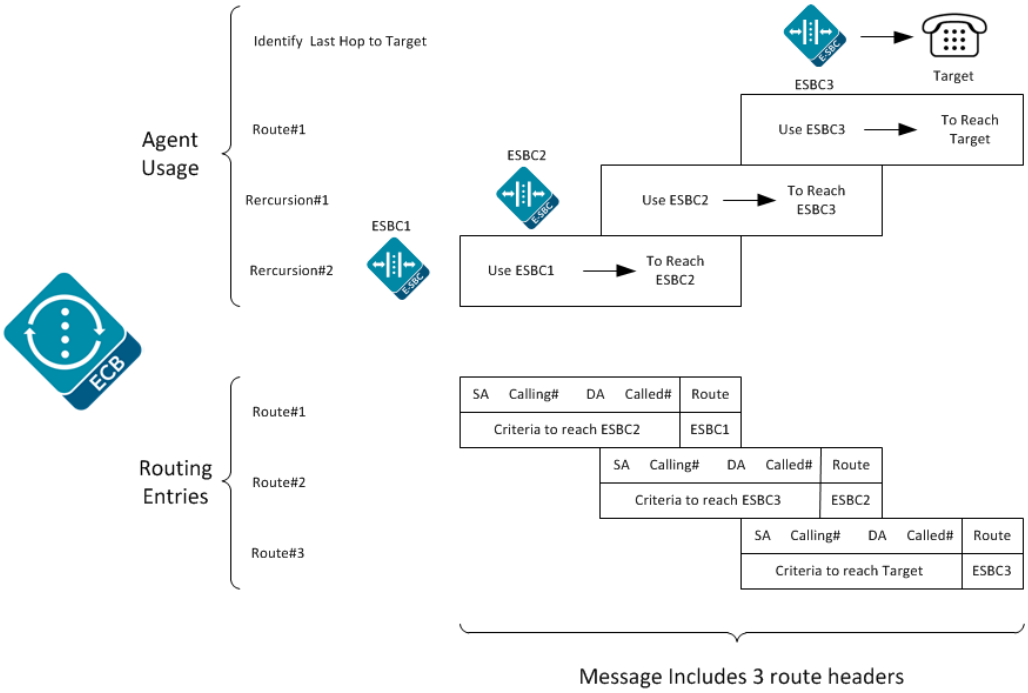
The Oracle Enterprise Communications Broker employs a brand new, purpose-built SIP routing engine for packet processing. Unlike traditional SIP proxies, application servers, or Session

Border Controllers, the Oracle Enterprise Communications Broker may be provisioned with a complete network topology map of all signaling entities, and use this provisioned data to make fully-informed routing decisions on how signaling flows should travel through a SIP network. Not satisfied with simply choosing a next hop and pushing the signaling message on its way, the Oracle Enterprise Communications Broker will look at the entire path from origin to destination, to find the path with the least cost, fewest active sessions, least number of hops, and so forth, and pre-populate the egress signaling message with a specific *route set* to inform each receiving device on the next element in sequence.

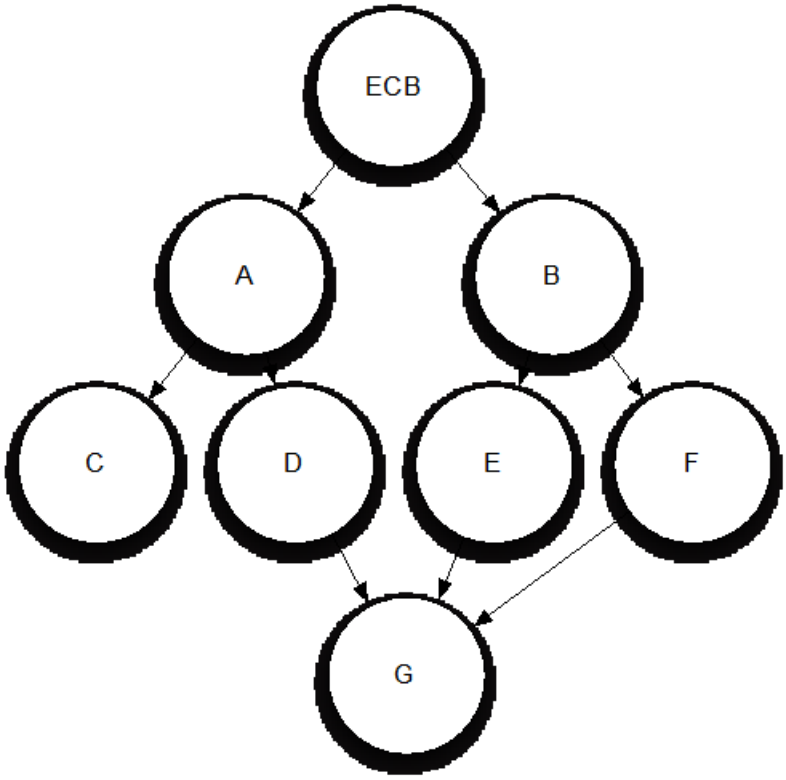
Recursive Routing

Conceptually, the Oracle Enterprise Communications Broker's routing engine is similar to a layer 3 router's recursive routing engine. The system provides the route engine with input criteria, including calling number, called number, "source agent", and "destination agent". The routing engine returns a set of results based on the lookup. Each result is processed recursively until complete, with each loop building another hop on the route.

The process of using recursion to create routes consists of identifying individual hops for an end-to-end path beginning with the last hop before the destination. The routing engine selects subsequent elements, (agents), to identify the hop that is penultimate to the previously identified hop until it has a full path between itself and the UE. The Oracle Enterprise Communications Broker's routing engine performs this process for all possible paths to each agent, creating multiple choices for the system to use as a route set for every individual call.



Consider the following diagram of a rudimentary network topology as an example.



First, the Oracle Enterprise Communications Broker finds that Agent G is the last stop on the signaling message's path. Next, resolves each way that it can reach Agent G. It finds that G is reachable via agents D, E, and F. Next, it looks up how to reach D, E, and F, and the route table yields Agents A and B. Because Agents A and B are directly connected to the Oracle Enterprise Communications Broker, route path identification is complete.

Identifying Contacts and Specifying Routes

Having determined the target for a call, the Oracle Enterprise Communications Broker creates route sets to the target. In addition, the Oracle Enterprise Communications Broker also finds all of a target's contacts and builds route sets for each of them. Depending on deployment and configuration, these contacts are available from multiple sources, both internal and external to the Oracle Enterprise Communications Broker. These sources provide the Oracle Enterprise Communications Broker with the agent of each contact, from which it can build routes to the contacts.

The Oracle Enterprise Communications Broker uses called, calling number and source agent from the source of the call, and the called number and destination agent to create a route. It goes about collecting this detail for target and all contacts using the following procedure:

1. Find the AoR and all associated contacts in the registration cache. Store agent(s) for route creation.
2. Find caller and callee's source context, as presented earlier in this document.
3. Convert user portion of request-URI and From URI to universal number via the dial plan and source context. These become called and calling number.
4. Lookup called and calling number in the user database. Retrieve each number's agent for route creation.

5. If the user database lookup does not produce source and destination agent, request this information via the LDAP database. If necessary, the system converts the universal number so the lookup format is compatible with the LDAP database.
6. If the LDAP lookup does not return source and/or destination agent, the Oracle Enterprise Communications Broker uses the host portions of the Request-URI and FROM URI or inbound agent (agent on which the call was received) as agents.
7. For any AoR returned from LDAP, the Oracle Enterprise Communications Broker performs another lookup in the registration cache and creates routes for the AoR.
8. If the LDAP lookup identifies other contacts, the system passes those contacts through the registration cache to identify its agent and build the route set.
9. All routes are built and, depending on forking configuration, placed in order.

If any of these sources are not configured or operational, the Oracle Enterprise Communications Broker proceeds to the next source. Note that, if agents are found in the user database, the system does not perform an LDAP lookup.

> **Note:**
>
> The procedures associated with an LDAP resource are equivalent to those with an LST resource. These procedures are also exclusive; LST and LDAP resources cannot be used simultaneously for routing.

The system forwards the request based on the route list and the forking configuration. By default, the system performs serial forking to all contacts using route cost to establish the order. The system can also perform parallel forking, if desired.

Route Selection

After constructing routes that can be used for a call, the Oracle Enterprise Communications Broker often has multiple routes from which to pick. Cost calculations for each path identify the route that the system uses.

Given the example above, the Oracle Enterprise Communications Broker determined that an inbound message sent to Agent G has three potential route paths:

- A->D->G
- B->E->G
- B->F->G

Each connection (route) between agents in the Oracle Enterprise Communications Broker's routing table may be assigned a cost that represents the desirability of that link. The Oracle Enterprise Communications Broker sums up the total cost for each path and orders them from least to highest cost. It then selects the least cost route set and forwards the message.

Forking

Forking is a routing option available on the Oracle Enterprise Communications Broker that causes an INVITE to be directed to multiple targets. There are a variety of types of forking that control the operational aspects of the function, such as timing and target lists. The Oracle Enterprise Communications Broker performs serial forking to all targets by default. The user

can globally configure the Oracle Enterprise Communications Broker to perform parallel forking.

The Oracle Enterprise Communications Broker learns of a user's multiple contacts from:

- A configured LDAP server
- The local registration cache
- The User Database

If there are multiple routes to contacts, the Oracle Enterprise Communications Broker uses its cost configuration to determine route preference. The system only uses backup routes if there is no response from the primary routes.

Parallel forking directs the INVITE to all of an AOR's contacts simultaneously. When any target responds, the Oracle Enterprise Communications Broker issues a CANCEL to the rest, ignoring any responses from them. Should the Oracle Enterprise Communications Broker receive error messages from all contacts, it provides the lowest number message back to the caller.

The user enables parallel forking via the **SIP Settings** control under the General icon. When enabled, the Oracle Enterprise Communications Broker directs all INVITEs to all of an AOR's contacts for every session.

Fork Groups

Fork-groups on the Oracle Enterprise Communications Broker are sets of one or more contacts that the system attempts to reach simultaneously. The system uses fork group order to specify when it tries to reach each fork group's contacts. This results in a hybrid of serial and parallel forking operation. The user can configure fork groups on agents, the registration cache and within the LDAP database. The user can also configure a global fork group timer with a value from 0 to 32 seconds on the sip-interface. If the system does not receive a response from any contact within that time, it tries the next fork group. Parallel forking must be enabled.

By default, the Oracle Enterprise Communications Broker assigns all contacts to fork group 1 and attempts to contact them serially, using the order in which it learns them. If desired, the user can enable parallel forking. By itself, parallel forking causes the system to attempt to reach all contacts simultaneously. Fork groups refine parallel forking, allowing the system to try all contacts in a group, and then move on to the next group.

The user names fork groups using decimal numbers between 1 and 100. This naming defines fork group order, with the system using fork group 1 first. The user configures objects with fork group numbers, based on a forking plan they devise.

The user can also configure a lookup query to LDAP databases to retrieve individual contacts' fork groups. The user must have previously modified the LDAP database to include a custom fork group field in contact records.

A use case for this feature could include the system attempting to reach a user's BYOD and desk phone simultaneously, then forwarding to an enterprise-preferred voicemail server if neither answers. For this to work, the BYOD and desk phone would be in the same fork group. The voicemail server would be a member of a higher numbered fork group. To ensure this order, the system assigns lower numbered fork groups with a higher precedence.

After establishing a session, other contacts may respond to try and start the session themselves. The Oracle Enterprise Communications Broker replies to these messages with a CANCEL.

Fork group operation does not exclude the use of primary and backup routes. The Oracle Enterprise Communications Broker still creates route sets for all contacts. If a contact fails via a primary route, the system attempts to reach the contact using all backup routes, based on cost.

If the Oracle Enterprise Communications Broker receives a redirect from an endpoint, the system adds the redirect target to the current fork-group and tries to contact it before attempting the next fork-group. If the global fork group timer expires before the system receives a redirect, however, the system proceeds to the next fork group.

The flexibility inherent in fork group operation requires the user to carefully plan forking prior to configuration. For each call, the system creates an ordered contact list, based on fork group configuration. Because the fork group assignment may affect multiple contacts, such as agent configuration, the user must be careful not to configure a sequence that would adversely affect calls to different end stations behind that agent.

Fork Group Assignment

The user configures fork groups to specify call attempt order for a given call. The Oracle Enterprise Communications Broker creates these call attempt lists based on each contact's fork group assignment.

Upon configuration, the system assigns fork groups to target endpoints as follows:

- User database—Each user database entry is assigned to the home-agent's fork-group.
- Registration cache—Each registration cache entry is assigned to the SIP registrar's fork-group.
- LDAP server—Each contact retrieved from an LDAP server is assigned to the a fork-group specified in the server's user record. If no fork-group was configured for the user in the Active Directory, the system assigns the target endpoint to the fork-group of the user's home-agent, as configured on the LDAP server.

Recall the contact source order used by the Oracle Enterprise Communications Broker:

1. Registration cache contact(s)
2. User database contact
3. LDAP contact(s)
4. LDAP AORs generating subsequent contact dips for additional registration cache contact(s)

By default, the Oracle Enterprise Communications Broker collects contacts from these sources and creates a contact list that follows the order in which the system learns them. This behavior is in accordance with default fork group operation, wherein all contacts are in fork group 1 and the mode is to fork serially only. As soon as the system finds differentiation between contact fork groups, however, it arranges contact lists using the fork group order.

Additional Targets

The user may want to include forking targets to stations that are not resolved as original call targets. Examples of these scenarios include directing calls to a preferred enterprise voice mail server if they are not picked up. The Oracle Enterprise Communications Broker provides for this using Additional target configurations. The user manually configures these devices within **Additional target groups**, which include one or more end stations. Agent and registrar configuration allows the user to select these groups as additional forking targets for all calls that use that agent or registrar's entries.

An additional target group is a list of agents (or endstations) that the Oracle Enterprise Communications Broker uses as candidates for either parallel or serial forking. The user configures these groups with fork group numbers, which the system then uses to define fork group order. The system adds additional target contacts to the forking sequence the same way it adds contacts for other objects with fork group configurations.

Configuring Fork Groups

Fork Group configuration requires that the user establish a clear plan prior to any configuration. Configurations established by this planning may include:

- The user may identify or create new agents as fork group targets.
- The user may identify usage and precedence policy for forking via the Registrar.
- The user may adjust fork group identification and precedence based on preferred LDAP lookup scenarios.

Coordinating the use of these sources and configuring the applicable objects establishes and refines fork group configuration. Applicable configuration objects include:

- Agent—The user creates new agents specifically for use in a fork group, or uses existing agents. The user configures an agent with a single fork group number, which the system applies to every call using that agent as a route.
- Additional targets—The user creates sets of targets to manually establish forking targets.
- Registrar—The user sets the registrar to a single fork group, which the system applies to every contact in the registrar.
- LDAP—The user defines a lookup query that pulls the pre-configured fork group assignment defined for the queried contact. The query must pull this fork group assignment from a custom attribute established on the LDAP database.

Configuring Fork Groups on Agents

The **Configuration tab** > **Agent** navigation sequence brings the user to the **Configure Agent** list. Each list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.

1. Click **Add** to create a new agent or **Edit** to add the agent to a fork group.

   The system displays the **Modify Agents** dialog.
2. **Additional target group**—Select and existing target group from the drop down list.
3. **Fork group**—Enter a digit (1-100) to specify this agent's fork group assignment.

Configuring Additional Target Groups

Additional targets are agents (or endstations) that are not contacts already targeted by a given call.

The user assigns additional target groups on a per-agent and a per-registrar basis.

The **Configuration tab** > **Agents** > **Additional Target Group** navigation sequence brings the user to the **Additional Target** list. This list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.

1. Click the **Add** link.

   The system displays the **Add Additional Target** dialog.

2. **Name**—Enter a name for this target. Use this name to assign this group to an agent or the registrar.
3. Note the Additional target group list.

   This list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.
4. Click the **Add** link.

   The system displays the subsequent **Add Additional Target** dialog. This list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.
5. **Additional session agent**—Select and existing target group from the drop down list, or enter an IP address of a target station.
6. **Fork group**—Enter a digit (1-100) to specify this agent's fork group assignment.

Assign your Additional Target Groups to the appropriate agent(s) and/or the registrar.

Configuring Fork Groups on a Registrar

The **Configuration tab** > **Registrar** > navigation sequence brings the user to the Registrar configuration dialog, which includes the Fork Group configuration fields.

1. **Additional target group**—Select and existing target group from the drop down list.
2. **Fork group**—Enter a digit (1-100) to specify the fork group for every contact in the registrar.

Configuring LDAP for Fork Groups

This procedure assumes the user has already defined and populated the custom LDAP attribute for specifying a user's fork group.

The **Configuration tab** > **LDAP** > navigation sequence brings the user to the LDAP server list. The list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.

1. Click **Edit**.

   The system displays the **Add LDAP config** dialog.
2. Scroll to the **Lookup queries** list. The list includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links.
3. Click **Add** to create a new lookup query.

   The system displays the **Add lookup query** dialog.
4. Referring to the **Fork group attribute** field, enter the name of the custom attribute in the LDAP database that includes fork group assignments.

Configuring the Global Fork Group Timer

The **Configuration tab** > **SIP Interface** > navigation sequence brings the user to the SIP Interface configuration dialog, which includes the global Fork Group timer configuration field.

1. **Fork group timer**—Enter a time in seconds (0-32) to specify the timeout the system uses to wait for responses from a fork group before it begins to try contacts the 'next' fork group. The default is zero (0). When set to default, the system waits for the standard SIP INVITE transaction timeout to expire before proceeding with the next group.

   After this timeout, the system drops responses received from contacts in the expired fork group.

Routing and ENUM

The ENUM functionality lets the Oracle Enterprise Communications Broker make an ENUM query for a SIP request. The ENUM lookup capability lets the Oracle Enterprise Communications Broker transform E.164 numbers to URIs during the process of routing (or redirecting) a call. During the routing of a SIP call, the Oracle Enterprise Communications Broker determines if an ENUM query is required and if so which ENUM server(s) need to be queried. A successful ENUM query results in a URI that is used to continue routing or redirecting the call.

Refer to the chapters on Agent and Route configuration for instructions on the related fields.

Route Types and Precedence

There are three types of routes used by the Oracle Enterprise Communications Broker. These include configured, default and implicit. The Oracle Enterprise Communications Broker uses these types, in conjunction with route cost, to determine route order. You create both configured and default routes in your route table. A default route is simply a route configured with wildcards for called number, calling number, source agent and destination agents. The system installs implicit routes dynamically when there are no explicitly configured routes to an agent. The system assumes the agent is to be a directly connected next-hop, and subsequently relies on the network infrastructure to reach that agent when needed.

The system orders routes by cost first, with the lowest cost being preferred. If costs are equal, the system orders by type, with the preference given to configured, then implicit and then default. If route cost and type are all equal, the system orders routes according to hop count, with the lowest number of hops being preferred.

Refer to the chapter on Route configuration for instructions on the related fields.

Active Directory and Oracle ECB Routing

A large percentage of Enterprises currently use call servers with Active Directory (Domain Controller) such as Media Servers, Exchange Servers, Lync Servers, etc. For Enterprises that integrate these servers in parallel to their existing communications infrastructure, or transition from their legacy Private Branch Exchange (PBX) to these types of servers, Active Directory becomes a more efficient and cost-effective way of routing the incoming calls within the core Enterprise network.

Clients using Microsoft servers such as a Lync Server deploy their own URI. Therefore, a user in a network with both a desk phone and a Lync client have an IP PBX extension/URI for the desk phone, and a different URI for the Lync client. Currently, all PSTN traffic is sent by default, to a legacy PBX in the core network. If the PBX does not recognize the extension/URI, the PBX forwards it to the Lync client. Sending traffic to the PBX first and then to the Lync Server can be costly in terms of compute resources and/or licensing fees. Routing all incoming sessions from a SIP trunk to the Lync Server first and then to a PBX can be costly.

As a solution, the Oracle Enterprise Communications Broker initiates a query to the Active Directory to determine how to route the call. The data fetched is the agent of the target(s), pre-configured in the database.

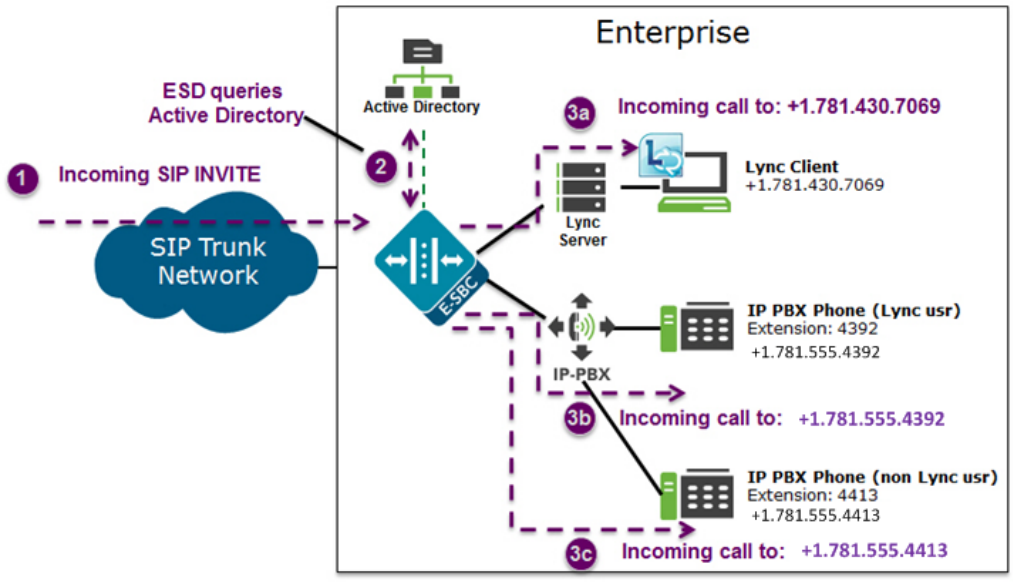
The Oracle Enterprise Communications Broker then stores data used to facilitate the routing decision of the call (performed by Lightweight Directory Access Protocol (LDAP) and then routes the call the first time to the applicable destination (PBX or Lync Server).

In scenarios where a user has multiple contacts such as both a Lync phone and a legacy PBX phone, calls destined for the Lync phone number can be routed to the PBX phone number, or calls destined for the PBX phone number can be routed to the Lync phone number. The destination is dependant on the current Oracle Enterprise Communications Broker configuration. The Oracle Enterprise Communications Broker uses the information stored in the Enterprise's Active Directory, compares it to the Oracle Enterprise Communications Broker configuration and then determines which phone number to utilize for the destined user.

> **Note:**
>
> The Active Directory-based call routing feature supports confidential and secure LDAP traffic support by using SSL/TLS (LDAPS).

Active Directory-based call routing is a feature of the Oracle Enterprise Communications Broker that facilitates the routing of incoming calls to the appropriate destinations within the Enterprise core network. The Oracle Enterprise Communications Broker's LDAP query to the Active Directory yields the agent of the phone number.

When the Oracle Enterprise Communications Broker receives an inbound SIP INVITE over a SIP Trunk (1), it checks the current LDAP configuration in the Oracle Enterprise Communications Broker. Depending on this configuration, the Oracle Enterprise Communications Broker then accesses the Enterprise's Active Directory to search for the applicable number being called via an LDAP query (2). When the query has found the agent of the called number, the Oracle Enterprise Communications Broker builds a route set for the call and routes the call, per the routing engine, directly to the call server client (3a) or to the IP PBX phone (3b) and 3c) as shown in the illustration below.



The Enterprise is responsible for migrating phone numbers from the legacy PBX to the call server by making the necessary updates in their Active Directory in order for the Oracle Enterprise Communications Broker to route the call properly. In the illustration above, the IP PBX extension (4392) is the primary telephone number (+1.781.555.4392); a secondary transition number (+1.781.430.7069) is assigned to Lync.

LDAP and Oracle ECB Routing

Lightweight Directory Access Protocol (LDAP) is the Protocol that the Oracle Enterprise Communications Broker uses to perform queries to the Enterprise's Active Directory to determine where to route incoming calls (to the call server or the IP PBX) in the Enterprise network. Session requests and responses are sent/received based on the Oracle Enterprise Communications Broker's LDAP routing configuration. LDAP determines the destination (call server user or non-call server user) and forwards the call accordingly.

The Oracle Enterprise Communications Broker, using LDAP, performs the following on an inbound call:

- Creates an LDAP search filter based on the dialed number and the configured LDAP attributes.
- Sends an LDAP search query to the configured LDAP Server.
- Creates a route list based on the query response(s) received from the LDAP Server and the applicable attributes it already has (caller number, callee number, caller agent).
- Routes calls based on the route list and routing order. The routing order is dependent on the LDAP attribute configuration and/or whether there was an exact match for the dialed phone number in the Enterprise's Active Directory.
- If configured, searches for additional AoR matches in Active Directory so that it can create additional routes to target users that have contacts stored in separate records.

To use AD as a source for home agent(s) names, the user creates lookup queries from the LDAP routing configuration dialogs. The Oracle Enterprise Communications Broker uses LDAP to retrieve that information and create routes. If the system cannot derive a home agent name from the query results, it routes the call to the configured default home agent.

> **Note:**
>
> The user must ensure that phone numbers in the LDAP database are unique. If the Oracle Enterprise Communications Broker encounters multiple records with the same number, the lookup fails.

The Oracle Enterprise Communications Broker keeps a permanent LDAP session open to all configured call servers. It sends an LDAP bind request on all established connections, to those servers. The first call server is considered the primary LDAP Server, and all others are secondary LDAP servers. If a query request sent to the primary server fails, the Oracle Enterprise Communications Broker sends the request to the next configured LDAP Server, until the request is successful in getting a response. If no response is received by the Oracle Enterprise Communications Broker and the Oracle Enterprise Communications Broker cannot find another route successfully, (all Oracle Enterprise Communications Broker configured attributes have been exhausted (local policies, policy attributes, etc.), it sends a busy to the caller.

LDAP performs call routing based on LDAP attributes configured on the Oracle Enterprise Communications Broker. The **route-mode** attribute setting determines how LDAP handles the called number when accessing the Enterprise's Active Directory. Routing modes can be set to any of the following:

- Match-only (default)

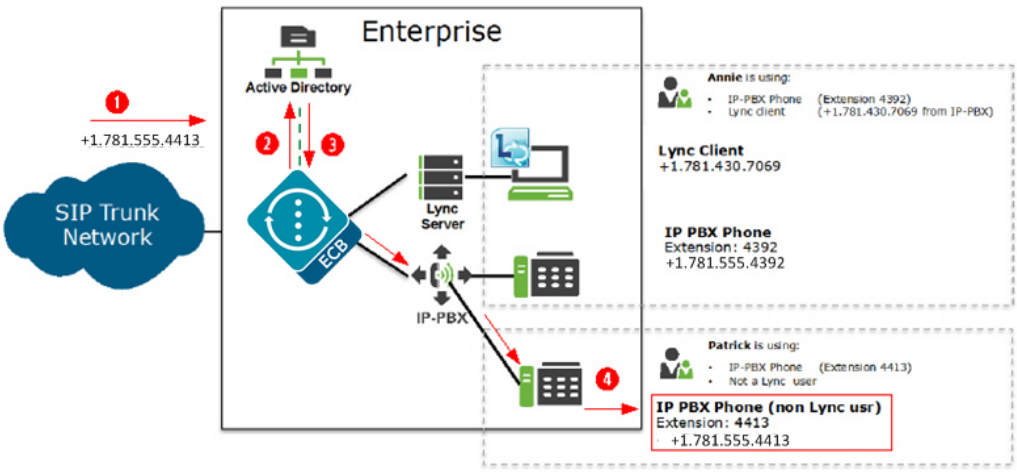
- Attribute-order
- Match-first

The following paragraphs describe each of these route-modes.

Match-only

If the LDAP **route-mode** attribute is set to match-only, the Oracle Enterprise Communications Broker performs as follows.

The Oracle Enterprise Communications Broker receives an incoming call to the Enterprise network. If the LDAP route-mode attribute on the Oracle Enterprise Communications Broker is set to match-only, LDAP queries the Active Directory to find the number that matches exactly to the incoming number. If the number is found, the Oracle Enterprise Communications Broker retrieves the entries' agent and builds a route list for that call.



Number	Description
①	Call comes into the Enterprise network (+1.781.555.4413)
②	Using the configured route-mode of match-only, LDAP queries the exact matching number in the Enterprise's Active Directory.
③	The Active Directory finds the matching number and that number's agent is included in the response to the LDAP query.
④	The Oracle Enterprise Communications Broker creates a route set for the call and forwards the call towards the destination phone number (same number as the number that initially called into the Enterprise in Step 1 (+1.781.555.4413)).

Attribute-order

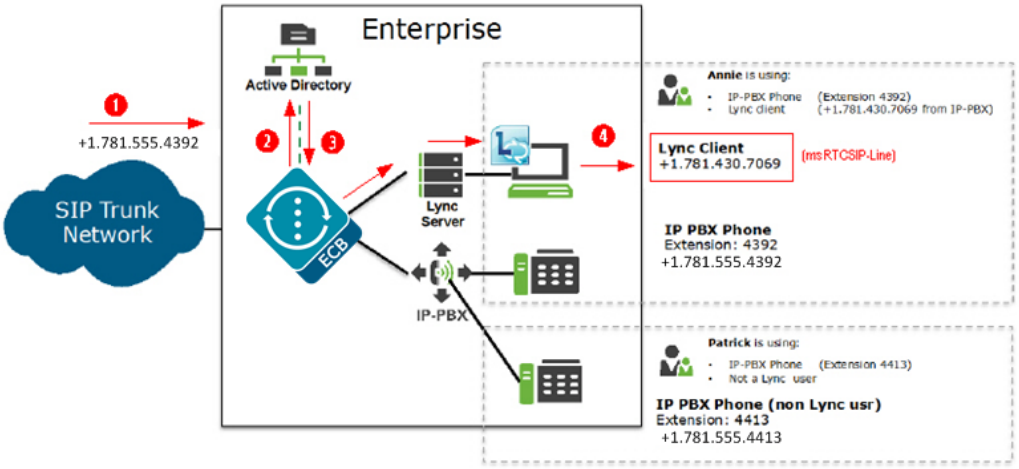
If the LDAP **route-mode** attribute is set to attribute-order, the Oracle Enterprise Communications Broker performs as follows.

The order in which the LDAP attributes are configured on the Oracle Enterprise Communications Broker determines the priority of each route. If an incoming call is destined for the IP-PBX , but the attribute name for a Lync client is configured first, the Oracle Enterprise Communications Broker uses the corresponding agent (Lync Server) to create the first route in the route list.

An entry in an LDAP search response must have at least one attribute that it matches in the Active Directory.

For example, the incoming phone number could be +1.781.555.4392 (which matches the IP-PBX phone number), and the attribute name msRTCSIP-Line (Lync attribute) in the response could be +1.781.430.7069 (Lync phone number). A route is created for the Lync phone number, even though the incoming phone number matches the IP-PBX phone number, since the msRTCSIP-Line attribute was configured first. Therefore, the Oracle Enterprise Communications Broker forwards the call to the Lync destination.

Likewise, if an Enterprise uses the same phone number for both Lync and IP-PBX phones, and the attribute-name msRTCSIP-Line is configured first (a Lync attribute), the Oracle Enterprise Communications Broker uses the corresponding agent (Lync Server) to create the first route in the route list.



Number	Description
①	Call comes into the Enterprise network (+1.781.555.4392)
②	Using the configured route-mode of attribute-order, LDAP queries the Active Directory for the agent of the matching number.
③	The Active Directory responds with the agent associated with the first configured LDAP attribute (+1.781.430.7069). In the illustration above, the number was associated with a Lync Client (msRTCSIP-Line) that was configured first in the LDAP configuration.
④	The Oracle Enterprise Communications Broker forwards the call to the applicable destination phone number's agent from the Active Directory response. (+1.781.430.7069).

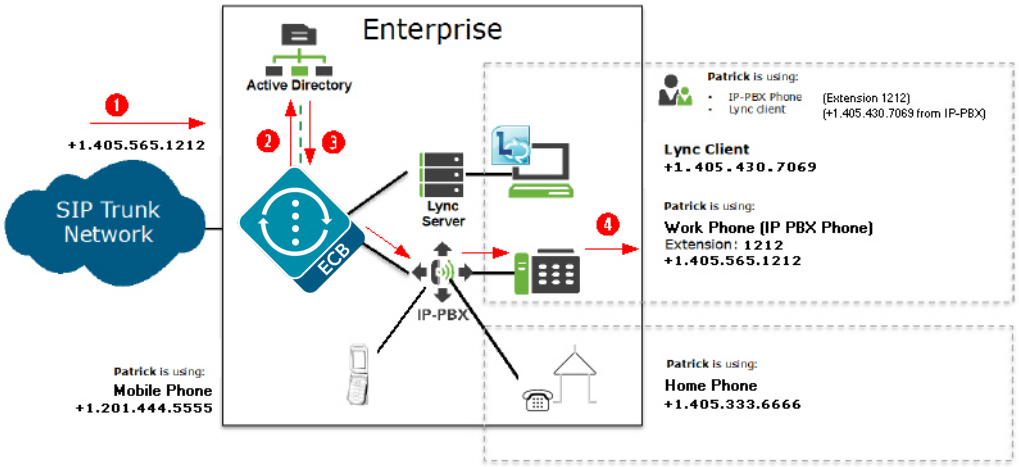
If you configure the attribute name msRTCSIP-Line first, the Oracle Enterprise Communications Broker uses the corresponding next hop (Lync Server) to create the second highest priority route in the route list. For example, the dialed telephone number could be +1.781.555.4392 (IP-PBX phone number), and the attribute-name msRTCSIP-Line in the response could be +1.781.430.7069 (Lync phone number). A route is created for the Lync phone number, even though the dialed telephone number is the PBX phone number.

Match-first

If the LDAP **route-mode** attribute is set to match-first, the Oracle Enterprise Communications Broker performs as follows.

When the LDAP query is sent to the Active Directory, the first exact match of the incoming phone number that the LDAP query finds in the Directory, is the number whose corresponding route gets the highest priority in the route list. For all other routes configured on the Oracle Enterprise Communications Broker, the ordering of LDAP attributes in the LDAP configuration determines the priority for each route.

For example, if the incoming number is +1.405.565.1212, and the Active Directory includes a configured mobile number first (+1.201.444.5555), a home number second (+1.405.333.6666) , and a work number third(+1.405.565.1212), the LDAP query searches the mobile number first, then the home number, then finds the exact match on the work phone number. The Active Directory responds with the agent information for the work phone number and the Oracle Enterprise Communications Broker creates a route list with this exact phone number, and then forwards the call accordingly.



Number	Description
①	Call comes into the Enterprise network (+1.405.565.1212)
②	Using the configured route-mode of match-first, LDAP queries the Active Directory for the agent of the matching number.
③	The LDAP query searches throughout the Active Directory until it finds the first exact match on the number. Active Directory responds with the exact phone number associated with the incoming number (+1.405.565.1212). In the illustration above, the number was associated with the work phone.
④	The Oracle Enterprise Communications Broker forwards the call to the agent of the applicable destination phone number from the Active Directory response. (+1.405.565.1212).

LDAP Messages

If LDAP message logging is enabled in the Active Directory, the Oracle Enterprise Communications Broker sends LDAP messages to a message log called sipdldap.log. This log records all received and sent LDAP messages. Messages are in ASCII encoded binary format.

Additionally, when LDAP is invoked for routing, the LDAP messages display in the GUI under the Monitor and Trace tab.

> **Note:**
>
> The Oracle Enterprise Communications Broker also supports transmitting LDAP messages using the IPFIX Protocol for the Palladion Mediation Engine.

LDAP Failure Events

If an incoming registration to a primary phone number in Lync fails, the phone number is routed to the IP PBX. If failures occur during LDAP queries for all LDAP Servers, the Oracle Enterprise Communications Broker logs the failure to the sipdldap.log, and proceeds with normal configured routing policies, if available.

> **Note:**
>
> The Oracle Enterprise Communications Broker always establishes the TCP/TLS connection towards the configured LDAP server(s). If a TCP connection fails, the Oracle Enterprise Communications Broker continues to attempt to re-establish the connection.

An LDAP connection failure can be due to any one of the following events:

- Oracle Enterprise Communications Broker receives a CANCEL message (LDAP connection termination). The Oracle Enterprise Communications Broker detects this if it receives or issues an 'unbind' operation. The session is then closed down at TCP/TLS.
- Oracle Enterprise Communications Broker receives a call failure message from Lync (TCP/TLS socket termination). If either side receives a finish message (FIN) or reset message (RST), the TCP socket closes per standard behavior, which triggers the LDAP layer to detect connection failure. The Oracle Enterprise Communications Broker fails over to a secondary LDAP Server, if configured; otherwise it periodically attempts to reconnect to the Primary LDAP Server.
- Oracle Enterprise Communications Broker is unreachable and SIP session towards Lync times out. User is enabled for Lync but the Lync Server is unreachable by the Oracle Enterprise Communications Broker so a timeout occurs. When consecutive LDAP queries timeout, the Oracle Enterprise Communications Broker concludes that the LDAP session has failed, and then proceeds to terminate the TCP/TLS connection.

The number of consecutive queries that timeout before a connection is considered failed, and the number of successive query timeouts for each LDAP Server can be set via configuration.

Oracle ECB Limitations using LDAP

The Oracle Enterprise Communications Broker uses LDAP in the Active Directory when determining the destination of incoming calls. However, the Oracle Enterprise Communications Broker has the following limitations when using LDAP:

- Supports LDAP sessions over the Oracle Enterprise Communications Broker media interfaces only (i.e., not on wancom0).
- Supports LDAPv3 only.
- Establishes a session over the following connections only:
  LDAP over TCP - default

LDAP over TLS (LDAPS)

Configuring LDAP for Routing

LDAP is the Protocol that the Active Directory uses for general interaction between and LDAP client and an LDAP server. You can configure the LDAP Server(s) in your network, and set the filters and the local policy that the LDAP Server uses when handling inbound Lync and PBX calls in the Enterprise core network.

You can use the following objects in the Web GUI to configure LDAP:

- LDAP Config—Configures the LDAP functionality on the Oracle Enterprise Communications Broker (i.e., name, state, LDAP servers, realm, authentication mode, username, password, LDAP search filters, timeout limits, request timeouts, TCP keepalive, LDAP security type, LDAP TLS profile, and LDAP transactions).
- Routing—Configures Active Directory attribute names, attribute format and regex extractions for routing SIP requests to the target's home agent. You configure this object for LDAP search queries in the Active Directory.
- Address of Record—Configures Active Directory attribute names, attribute format and regex extractions for identifying other addresses of record for the request URI and from. The Oracle Enterprise Communications Broker uses any AoR information provided by these queries to generate additional routes for the session, using the same process it used for the original request URI and from.

3
Registrar and Authentication

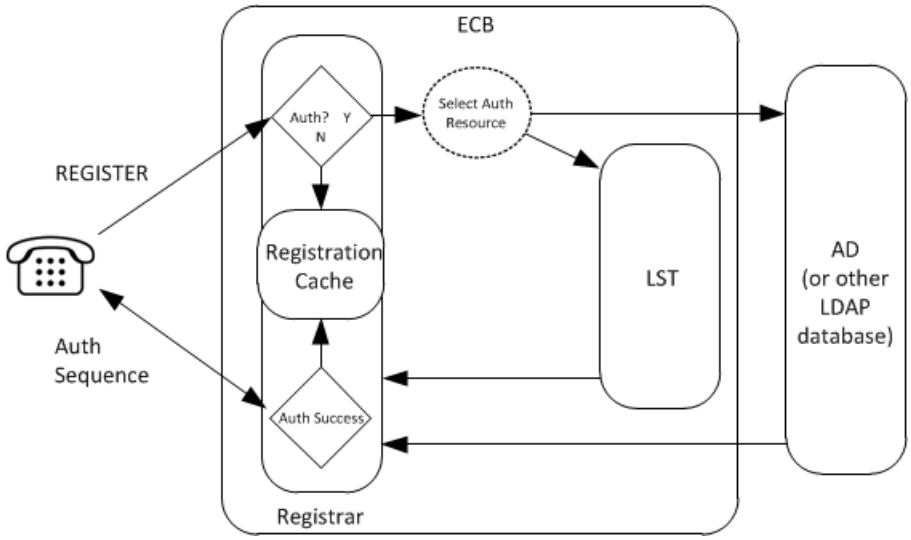
Registrar Function

By providing a location service from within it, the Oracle Enterprise Communications Broker offloads related infrastructure from providing that information for every session. The Oracle Enterprise Communications Broker can use SIP digest authentication to confirm service authorization and verify user registrations via internal or external mechanisms. If using an external mechanism for this purpose, some adaptation of that mechanism is required. The user enables the registrar, configures the applicable domains (including serviced and digest domains) and, if required, defines the authentication to be used for all registrations via configuration on the GUI.

The Oracle Enterprise Communications Broker's single registry service is enabled globally. When registration functionality is enabled, the Oracle Enterprise Communications Broker actually registers endpoints rather than only caching and forwarding registrations to another device.

On receiving a REGISTER message, the Oracle Enterprise Communications Broker checks if it is responsible for the domain contained in the Request-URI, as configured in the domains list. The Oracle Enterprise Communications Broker begins registrar functions for all requests that match a configured domain.

If there is no authentication configured, the system adds every user that attempts to register to the registration cache. If authentication is configured, the system can authorize/verify the user via the LST or via an external LDAP resource. In these cases, the system uses SIP digest to authenticate the user, based on authentication information from the LST or LDAP. Detail on authentication and interaction with LDAP resources, especially Active Directory, is presented below.



A UA is fully registered after the system installs it in the registration cache, after which the Oracle Enterprise Communications Broker sends a 200 OK message back to the registering UA.

When a user registers with the registrar, the system looks for the To header AoR in the LST. If the LST contains a subscriber with the AoR (or username if no AoR specified) that matches, the system adds the universal number of the subscriber as an alias to the registration cache.

Register Refresh

When a UA sends a register refresh, the Oracle Enterprise Communications Broker first confirms that the authentication exists for that UE's registration cache entry, and then is valid for the REGISTER refresh. (If a valid hash does not exist for that AoR, then the Oracle Enterprise Communications Broker sends a request to its source database (LST or LDAP) to retrieve authentication data once again).

Next, the Oracle Enterprise Communications Broker determines it can perform a local REGISTER refresh or if the source database needs to be updated. If any of the following 3 conditions exists for the re-registering UA, the Oracle Enterprise Communications Broker updates the database:

- The location update interval timer has expired—This value, configured in the sip registrar configuration element ensures that source database always has the correct Oracle Enterprise Communications Broker address by periodically sending request messages for each registered contact.
- The message's call-id changes while the **forward-reg-callid-change** option in the sip config configuration element is set. This covers the case where the UA changes the Oracle Enterprise Communications Brokers through which it attaches to the network.
- The REGISTER message's Cseq has skipped a number. This covers the case in which a user registered with Oracle Enterprise Communications Broker1, moves to Oracle Enterprise Communications Broker2 , and then returns to Oracle Enterprise Communications Broker1.

If the Oracle Enterprise Communications Broker updates the source database because of matching one of the above conditions, the access side expiration timer per contact is reset to the REGISTER message's Expires: header value, and returned in the 200 OK. This happens even in the case when the reREGISTER was received in the first half of the previous Expires period. In addition, the core-side location update interval timer are refreshed on both active and standby.

When the above three conditions are not met, the registration expiration proceeds normally.

If the timer has not exceeded half of its lifetime, a 200 OK is returned to the UA. If the timer has exceeded half of its lifetime, the Oracle Enterprise Communications Broker just refreshes the access-side expiration timer; the registration cache expiration timer for that AoR begins its count again.

Proxy Registrations

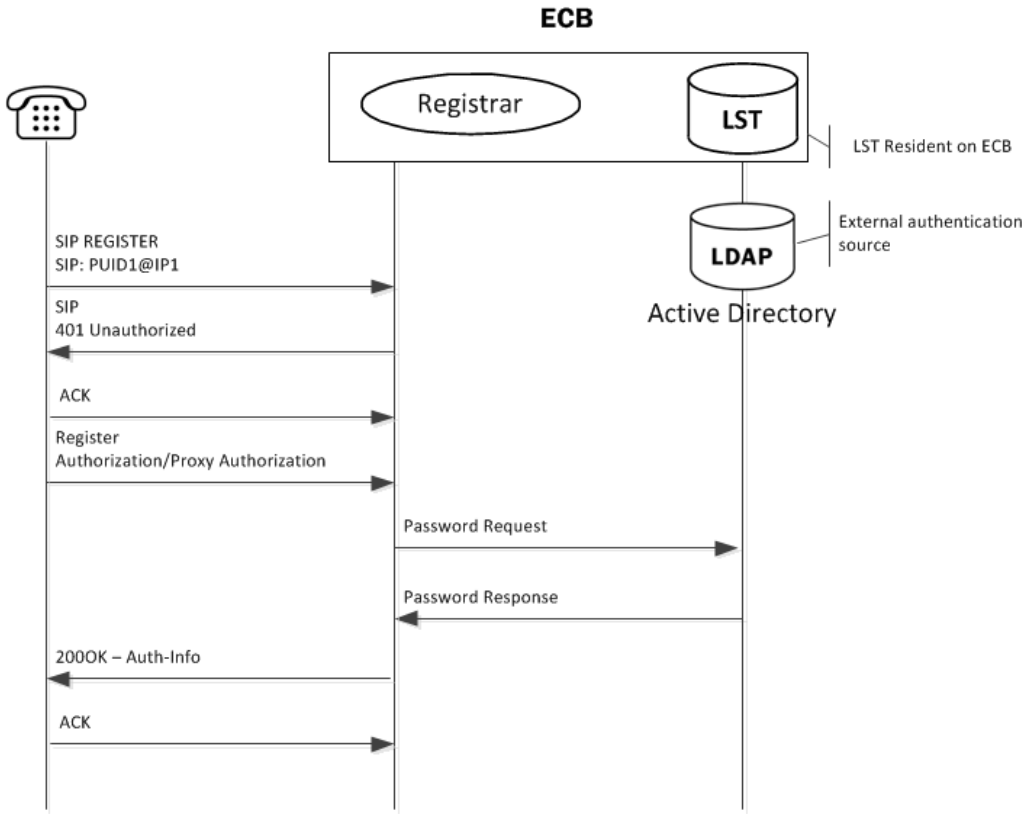
The Oracle Enterprise Communications Broker can proxy registrations when it receives REGISTERs for domains for which it is not a registrar. The user enables this functionality within the **sip-interface**. By default, the Oracle Enterprise Communications Broker rejects the registration.

The Oracle Enterprise Communications Broker's **sip-interface** configuration includes a checkbox titled **Proxy Registrations**, with which the user can enable this function. When checked, the Oracle Enterprise Communications Broker proxies the registration towards the intended registrar. When unchecked, the Oracle Enterprise Communications Broker responds with a **403: Unauthorized** message.

Message Authentication for SIP Requests

The user can configure the Oracle Enterprise Communications Broker to authenticate REGISTER requests. The Oracle Enterprise Communications Broker offers a single Registrar for location services on user-specified listed domains. Registration may or may not include user authentication. If it does, the user can select a local, text-based resource called the Local Subscriber Table (LST) as an authentication source. The user can also configure the Oracle Enterprise Communications Broker as an LDAP client, allowing it to perform LDAP-compliant processes and retrieve authentication information from an external resource, usually Active Directory. The Oracle Enterprise Communications Broker populates the registration cache with contacts for AORs upon successful authorization/authentication.

The Oracle Enterprise Communications Broker uses SIP digest authentication as a means of challenging an end station for applicable registration attempts. The diagram below presents the overall authentication/authorization sequence, including the Oracle Enterprise Communications Broker confirming the registration via an LST or an external LDAP server.



Authentication

To authenticate the registering user, the Oracle Enterprise Communications Broker needs the hash of the end station's password. It requests these from the local LST of an LDAP server by sending it an LDAP query for the configured field.

The hash consists of an MD5 hash made up of the following components:

```
MD5(username:digest-realm:password)
```

The transaction is conducted with the server defined in the Registrar configuration's credential retrieval method parameter. This parameter is populated with the name of the LDAP sever.

SIP Authentication Challenge

When the Oracle Enterprise Communications Broker receives a response from the HSS including the hash value for the user, it sends a SIP authentication challenge to the endpoint, if the endpoint did not provide any authentication headers in its initial contact the with Oracle Enterprise Communications Broker. If the endpoint is registering, the Oracle Enterprise Communications Broker replies with a 401 Unauthorized message with the following WWW-Authenticate header:

```
WWW-Authenticate: Digest realm="atlanta.com", domain="sip:boxesbybob.com",
qop="auth", nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE,
algorithm=MD5
```

Authentication Header Elements

- Domain—A quoted, space-separated list of URIs that defines the protection space. This is an optional parameter for the "WWW-Authenticate" header.
- Nonce—A unique string generated each time a 401/407 response is sent.
- Qop—A mandatory parameter that is populated with a value of "auth" indicating authentication.
- Opaque—A string of data, specified by the Oracle Enterprise Communications Broker which should be returned by the client unchanged in the Authorization header of subsequent requests with URIs in the same protection space.
- Stale—A flag indicating that the previous request from the client was rejected because the nonce value was stale. This is set to true by the SD when it receives an invalid nonce but a valid digest for that nonce.
- Algorithm—The Oracle Enterprise Communications Broker always sends a value of "MD5"

SIP Authentication Response

After receiving the 401/407 message from the Oracle Enterprise Communications Broker, the UA resubmits its original request with an Authorization: header including its own internally generated MD5 hash.

Authentication Check

At this point, the Oracle Enterprise Communications Broker has received an MD5 hash from the HSS and an MD5 hash from the UA. The Oracle Enterprise Communications Broker compares the two values and if they are identical, the endpoint is successfully authenticated. Failure to match the two hash values results in a 403 or 503 sent to the authenticating endpoint.

Retrieving Information from Active Directory

The Oracle Enterprise Communications Broker performs SIP Digest authentication against users attempting to register. It can use pre-configured information from Active Directory to perform such authentication. Access to Active Directory uses standard LDAP processes to retrieve the information needed and to offload the processing from other resources to the Oracle Enterprise Communications Broker.

The Oracle Enterprise Communications Broker can obtain registration authentication information directly from Active Directory when you modify the Active Directory schema to include the Oracle-specific attributes and object classes that the Oracle Enterprise Communications Broker needs to authenticate users..

The Oracle Enterprise Communications Broker operates by issuing LDAP requests from Active Directory for data from "password" attributes, using Active Directory's standard **sAMAccountName** to match the Request URI username to create new attributes in Active Directory. One of these attributes must be populated with the digest realm. A Dynamic Link Library (DLL) installed on Active Directory intercepts the password change hashes and writes them to another attribute. The DLL then creates a hash of the username, digest realm, and password hash to be returned to the Oracle Enterprise Communications Broker within the LDAP response. The Oracle Enterprise Communications Broker extracts the password hash, compares it to the hash provided by way of SIP digest, authenticates, and registers the user when there is a match.

- orclDigestRealmAttribute—Populated with digest realm.
- orclDigestPwdAttribute—Populated with hash of Active Directory password during each password change.
- orclAgentNameAttribute—Populated with user's agent for the purpose of routing. See Active Directory and Oracle ECB Routing in this document to understand how the Oracle Enterprise Communications Broker uses this attribute.

Oracle can provide the oidpwdcn.dll, scripts to create the needed attributes, scripts to populate the digest realm attribute, and a README.TXT with instructions on how to perform all procedures. Appendix C provides instruction on getting this methodology operational.

LDAP and Oracle ECB Authentication

Lightweight Directory Access Protocol (LDAP) is the Protocol that the Oracle Enterprise Communications Broker uses to perform queries to the Enterprise's Active Directory to validate registration attempts in the Enterprise network. Requests and responses are sent/ received based on the Oracle Enterprise Communications Broker's LDAP configuration. The Oracle Enterprise Communications Broker's LDAP client queries an LDAP server, usually Active Directory for password information for a user attempting to register. This request and response process verifies that the user can get registration servers (authorization) and verifies that the user is who they say they are (authentication). Once both these stages complete successfully, the Oracle Enterprise Communications Broker registers the user.

The Oracle Enterprise Communications Broker, using LDAP, performs the following on a registration attempt:

- Creates an LDAP search filter based on the dialed number and the configured LDAP attributes.
- Sends an LDAP search query to the configured LDAP server.

You configure LDAP servers and filters, on the Oracle Enterprise Communications Broker.

The Oracle Enterprise Communications Broker keeps a permanent LDAP session open to all configured call servers. It sends an LDAP bind request on all established connections, to those servers. The first call server is considered the primary LDAP server, and all others are secondary LDAP servers. If a query request sent to the primary server fails, the Oracle Enterprise Communications Broker sends the request to the next configured LDAP server, until the request is successful in getting a response. If no response is received by the Oracle Enterprise Communications Broker, it replies to the registering endpoint with a (401? authentication failure?).

Configuring LDAP for Authentication

LDAP is the protocol that the Active Directory uses for general interaction between and LDAP client and an LDAP server. You can configure the LDAP server(s) in your network, and set the filters and the local policy that the LDAP server uses when handling inbound Lync and PBX calls in the Enterprise core network.

You can use the following objects in the Web GUI to configure LDAP:

- LDAP Config—Configures the LDAP functionality on the Oracle Enterprise Communications Broker (i.e., name, state, LDAP servers, realm, authentication mode, username, password, LDAP search filters, timeout limits, request timeouts, TCP keepalive, LDAP security type, LDAP TLS profile, and LDAP transactions).
- SIP Authentication—Configures the Active Directory attribute names for the Oracle Enterprise Communications Broker's query-digest-username-attribute and digest-hash-attribute fields. These fields specify where the Oracle Enterprise Communications Broker verifies authentication attempts.

See the section on Active Directory and Oracle ECM Routing for important information about:

- LDAP messages
- LDAP failure events
- Oracle ECB limitations using LDAP

That information applies equally to the authentication functionality explained here.

4
Getting Started

Oracle® recommends that you review the topics in "Getting Started" before working with the system to ensure success with the tools and functions provided.

Browser Support

You can use the following Web browsers to access the Oracle Enterprise Communications Broker (ECB) Web GUI:

- Internet Explorer versions 9.0 and higher
- Mozilla Firefox versions 12.0 and higher
- Google Chrome versions 19.0.1084.46m and higher

> **Note:**
>
> After upgrading the software, clear the browser cache before using the ECB Web GUI.

Log On and Log Off

This section provides the concepts and procedures for logging on to and logging off from the Web GUI.

User and Administrator Access

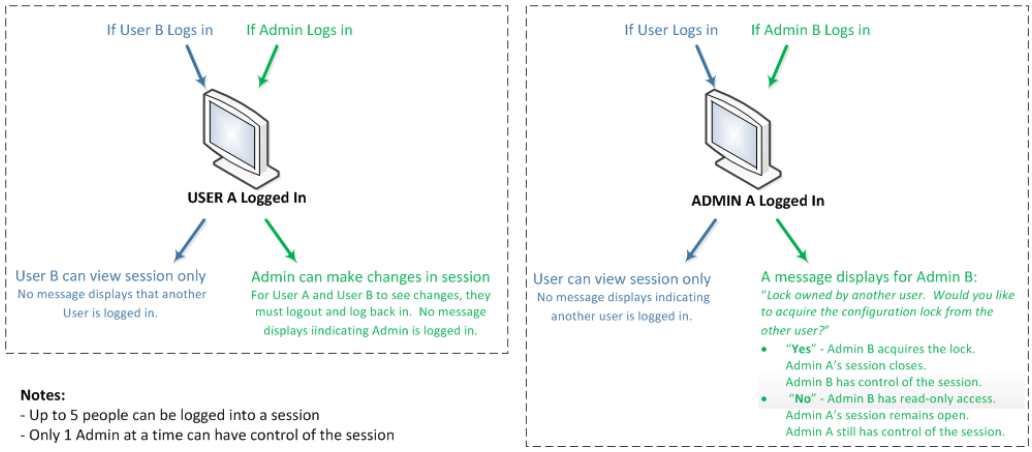
You can logon to the Web GUI using your Web browser. There are two types of user logons:

- **User** - Allows viewing (read-only) access to the Web GUI.
- **Administrator** - Allows Superuser access to the Web GUI.

For specific rules that apply to the User and Administrator when using the Web GUI tabs, see the respective topics.

Simultaneous Logons

The Web GUI allows simultaneous logons for both the User and Administrator. Session availability to the User and Admin depends on which type of user is logged onto the session. The following illustration shows a scenario of a User and an Administrator logged onto a Web GUI session.



Up to five users can log onto the same session at the same IP address at the same time. Only one Administrator at a time can have full control of a simultaneous session. If more than five users attempt to log on, the system displays the following error message:

```
User limit reached. Please try again later.
```

Radius Server in the Network

The Web GUI supports authentication functionality similar to a user logging on by way Secure Shell (SSH), and SSH File Transfer Protocol (SFTP).

The Web GUI supports RADIUS authentication. The following table describes the functions available to the Administrator and User levels.

User Class	Access
When you configure the RADIUS server as userclass=admin	the system allows the Administrator full access to all features and functions after logging onto the GUI.
When you configure the RADIUS server as userclass=user	the system limits User access to the following features and functions after logging onto the GUI. Full access to all SIP Monitor and Trace features and functions Can download the following files in System File Management: • Backup configuration • Configuration CSV • Local subscriber table (LST) • Log • Software image • SPL Plug-in (SPL)

Note:

A user with User privilege cannot upload files in System File Management.

Log On to the Web GUI

You can log on to the Oracle Enterprise Communications Broker (ECB) as a User or an as Administrator, depending on your permissions.

The system defaults for user name and password follow:

- User. The username is **user** and the password is **acme**.
- Administrator. The user name is **admin** and the password is **packet**.

If you previously changed the default password, use that one to log on.

If your system Administrator configured the optional log on page message, the system displays the message after you enter your logon credentials. After reading the message, click **Close** and the system displays the GUI.

1. On a PC, open a supported Internet browser.
2. Start the GUI with either the HTTP or HTTPS logon.

   ```
   http://<Server IP address>
   https://<Server IP address>
   ```

   > **Note:**
   >
   > Whether you log on using HTTP or HTTPS depends on the settings for your deployment. Contact your system Administrator for more information.

3. Enter your Web GUI username and password.
4. Click **Login**.

Log Off the Web GUI

To log off from the Web GUI, click **Logout** from the <logged-on-username> menu in the upper right corner of the Web GUI. In the following illustration, Admin is the name of the user who is logged on.

The system logs you off and displays the log on page.

Service Provisioning

After the Oracle Enterprise Communications Broker is operational, network architects and communications service provisioning technicians specify call services using the controls

available from the Service Provisioning icons. These controls, available from the Configuration tab, are defined in this section.

In contrast, System Administration controls, also available from the Configuration tab, specify how to manage the system itself and are documented in the *Oracle Enterprise Communications Broker System Administrator's Essentials Guide*.

Configuration Icons

The table below provides high-level descriptions of the Oracle Enterprise Communications Broker's Service Provisioning controls.

Icon	Description
Dial Plan	Add multiple dialing-contexts and dial-patterns. Dialing-contexts define the system behavior for calls placed to and from either a corporate of geographic focus. Dialing-contexts include multiple dial-patterns, which define the normalization required to most effectively manage diverse signaling structures.
Agents	Add agents. An agent is usually a SIP-aware device that serves as a transit target and/or source for signaling managed by the Oracle Enterprise Communications Broker. Agents are often specified as next-hops for the purposes of routing. Indirect agents, Oracle Enterprise Communications Broker route termination points that require further routing to reach an end station are also configured here. In addition, configuration used to access ENUM servers is performed here.
Users	Users - Add user and other key phone numbers associated with the enterprise. The user database serves as a directory for phone numbers that need communications services. This database can specify each entry's source context, which can provide a starting point for processing the logic behind a user's call treatment. It also can specify each user's agent, providing a physical location for routing user's calls.
Routing	Add service routes. Route-entries specify paths for signaling traffic, allowing you to specify policy and cost for traffic based on source and/or destination.

Web GUI Tools

The Web GUI provides some tools that apply to the entire GUI and other tools that apply to specific functions on a tab. For example, "Customizing the Page Display" applies to all pages and "Add widget" applies only to the Home page. Some tools are activated by icons and some are activated by links. The display of icons and links depends on whether the system displays Expert mode or Basic mode.

Global Tools

The following paragraphs describe the tools you can use to enhance your Web GUI experience. These tools apply to the Configuration, Monitor and Trace, and System tabs.

The User Menu

Oracle Enterprise Communications Broker dialogs include a User menu. This menu is located in the upper right-hand corner of each Oracle Enterprise Communications Broker dialog, and is labeled with the currently logged in user's name.

Commands the user can execute from the User menu include:

- Screen help
- Help topics
- About
- Logout

Help

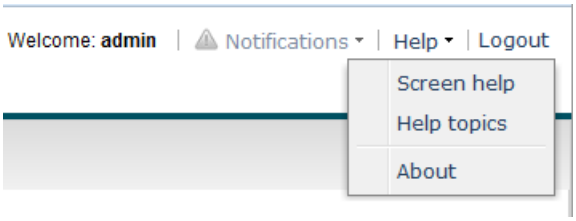
The logged on user button on the Web GUI displays the following information:

- **Screen Help**. Short descriptions of elements on the page.
- **Help Topics**. Online Help system containing topics about the tasks that you can perform on the Web GUI.
- **About**. Oracle notices and disclaimers, Oracle terms and restrictions, and third-party notices.

Screen Help

Screen Help provides an overlay on the current screen with pointers that indicate specific tasks you can perform. When you select **Help** > **Screen** help in the upper right corner of the page, an overlay displays with screen pointers to specific areas of the blurred-out screen. Clicking anywhere on the screen closes this help method.

You can display screen help on the main screens for each tab (Configuration tab (expert and basic modes), Monitor and Trace tab, and System tab).

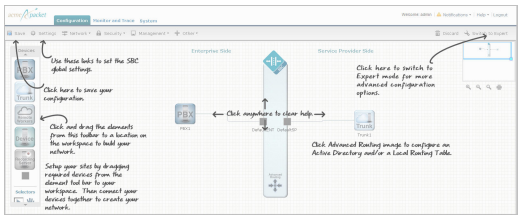


To display Screen Help:

1. Select **Help** > **Screen** help in the upper right corner of the screen. An overlay displays on the screen with help pointers to tasks you can perform.

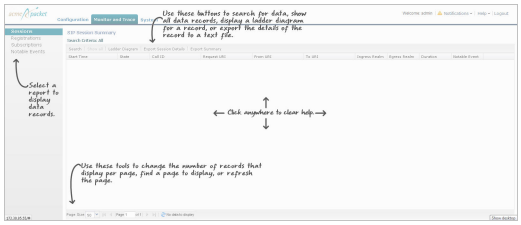
   The following illustrations show the screen help for each tab. If a User is logged into a session as “view-only”, some of the screen help pointers are not applicable.

   Configuration Tab (Basic mode)

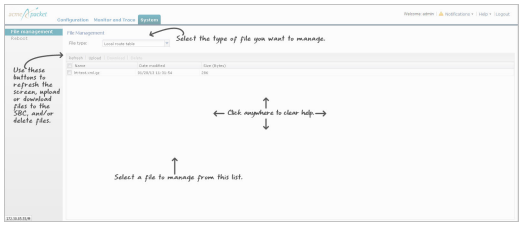


Configuration Tab (Expert mode)

Monitor and Trace Tab



System Tab

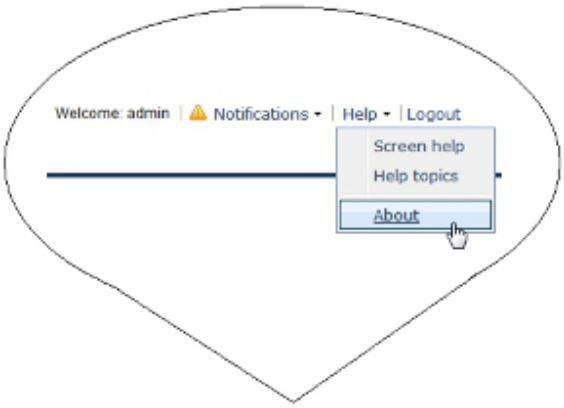


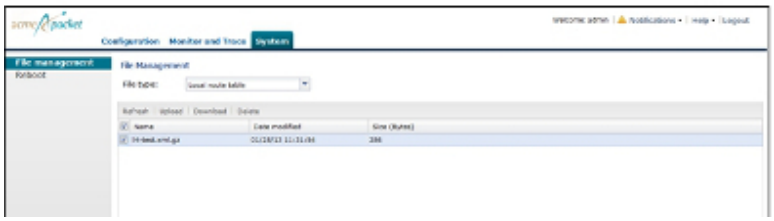
Help Topics

The Web GUI provides more detailed online help for the Configuration tab, Monitor and Trace tab, and System tab if required. You can select **Help** > **Help Topics** to display a menu of help topics you can click on to get more information about a topic. You can access help from any page in the Web GUI.

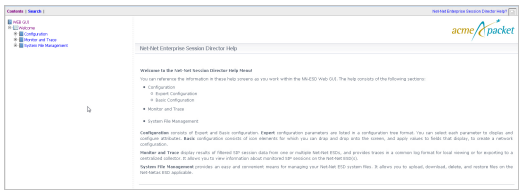
To display Help:

1. From any page in the Web GUI, select Help->Help topics in the upper right corner on the screen.





A new tab opens in your browser that contains a menu that provides help for the various aspects of your device.



2. Click on an element in the menu for help about that element
3. Close the tab by clicking the “x” in the upper right corner of the tab. Or drag the tab away from the browser to keep it open in a separate window while you continue to work in the Web GUI.

About this Product

To view information about this product, select **About** from the Help menu.

Procedure

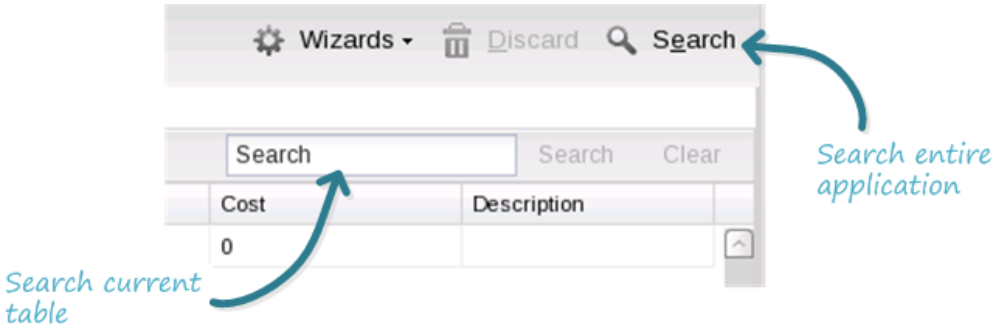
1. From the Web GUI, click **Help** > **About**.
2. Scroll to view the following information:
   - Platform type
   - Software version number
   - Legal notices
   - Copyright information
   - Open Source Mailing Address
   - Trademark recognition
   - Licensing information

The Search Tool

Search functionality is available on the Oracle Enterprise Communications Broker from both a system-wide and a dialog-specific perspective. Dialog-specific searches find text within the current dialog's table (grid).

The system-wide Search link, at the top right of all dialogs next to the magnifying glass icon, opens a follow-up dialog from which you configure your search and examine results. Text that can be found using search includes object names, attribute names and values. The search results window provides a link on the found object that takes the user to that object.

Table search is available on both the Users and Routing table configuration dialogs. To use, click inside the search field, type your string into the Search field and click the **Search** button. The system looks into the table and highlights matching text. There is also a **Clear** button that clears removes highlights.



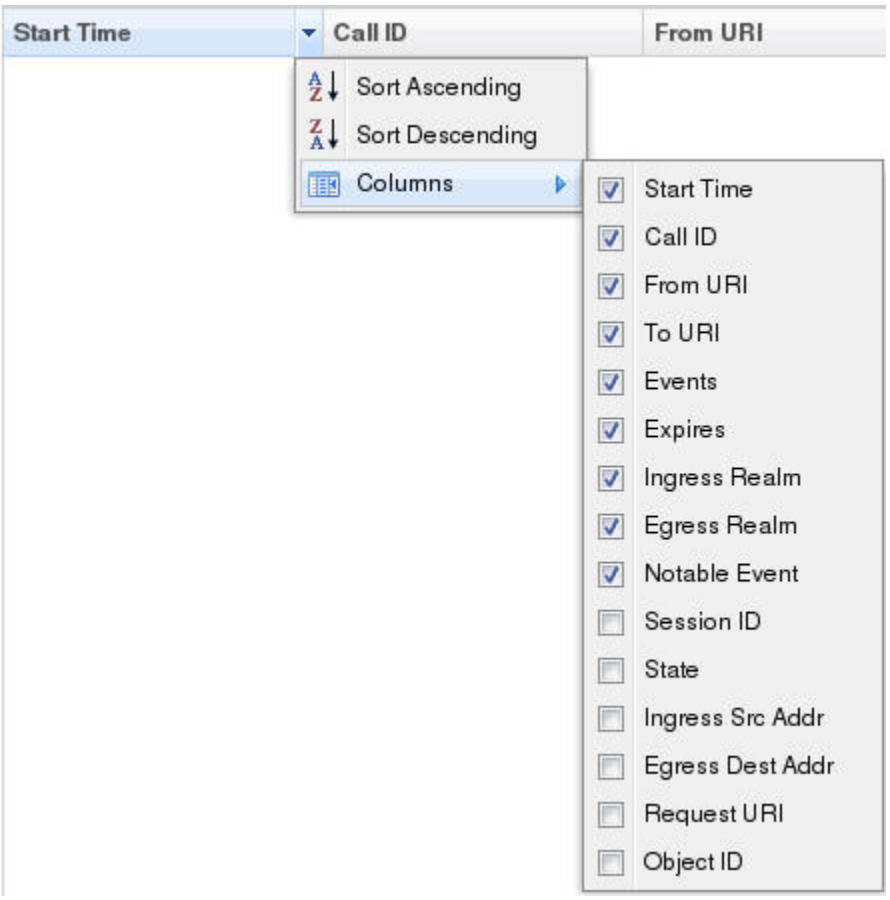
Customize the Page Display

You can customize the display of the data on Web GUI pages by selecting which columns display, the information type, and the sort order.

1. Place the cursor on a column heading.

   The system displays a down arrow in the column heading.

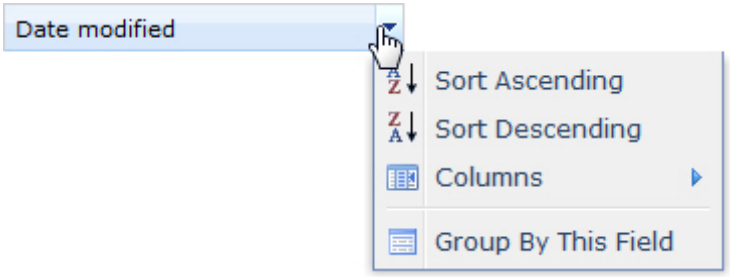
2. Click the down arrow to display the customization menus. For example,



Group by Field

The Group by This Field option is available from the System tab only.

When on the System tab, the Group by This Field option displays in the column drop-down box.



This option allows you to group items on a page according to the column heading you select.

Configuration Tools

The Web GUI provides specific tools within the Configuration tab that you use to configure the Oracle Enterprise Communications Broker. The following paragraphs describe these tools.

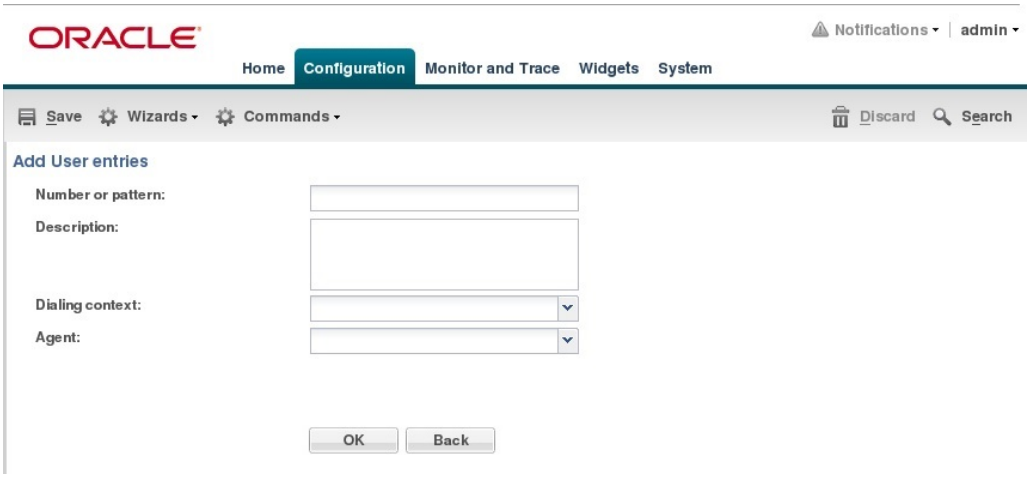
Web GUI configuration on the Oracle Enterprise Communications Broker is simple and consistent. Users familiar with GUIs will need little or no instruction on the intuitive and familiar controls, which include buttons, navigation links, text fields and drop-down lists.

Oracle Enterprise Communications Broker configuration on the Oracle Enterprise Communications Broker follows the same principle as you find on most devices with a GUI configuration tool. The configuration that resides on the Web GUI is not the same as the configuration on the device itself. Rather the Web GUI is a container for configuration changes you intend to make on the Oracle Enterprise Communications Broker. You invoke a two-step process, including Save and Activate, to transfer and your configuration changes to the Oracle Enterprise Communications Broker and then make them operational.

Configuration Tab Controls

The Configuration Tab provides a set of controls to aid you with performing configuration tasks and applying the configuration to the system.

The following illustration shows a typical configuration page and the controls that the system provides.



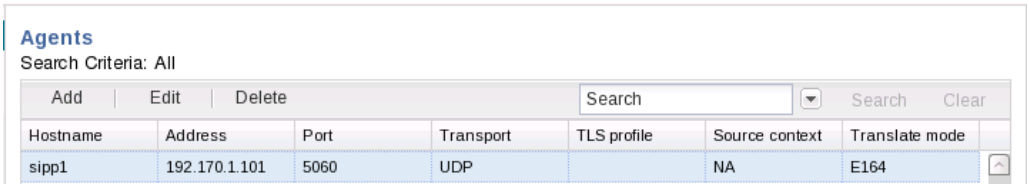
The following table explains the functionality of each control displayed on a configuration page.

Control	Effect
Notifications	Displays alerts about the state of the configuration, for example when you need to save changes or activate the configuration.
Admin	Displays links to screen help, help topics, about the Oracle Enterprise Communications Broker, and logout.
Save	Initiates Save and Activate process on your system. This process includes a configuration verification. The system displays any configuration errors list in a panel at the bottom of the page.
Wizards	Displays a menu of wizards for configuring global settings. The menu includes setting boot parameters, entitlements, initial configuration, license, login banner, time zone, and upgrade software.

Control	Effect
Commands	Displays a list of commands to show the state of the configuration. The menu includes links to the editing configuration, the running configuration, and the configuration version.
Discard	Discards all changes to all dialogs made prior to performing Save and Activate. The control becomes active when you click **OK** to accept configuration changes in a dialog.
Search	Displays the Search dialog where you can search the configuration by object name, attribute, or value.
OK	Accepts all changes made in the dialog, transfers the changes to the system, and returns to the previous screen.
Back	Returns the previous page. If you have made changes to any fields in the dialog, the GUI displays a confirmation dialog where you can save or discard changes to the current dialog.

GUI Configuration Editing Controls

The web GUI provides you with object editing controls that are dependent on the current dialog. The Agent list dialog, which includes the standard toolbar, is shown in the diagram below. You highlight an object to edit or delete it, or you click Add to create a new object.



You make the configuration changes themselves by typing your entries into text fields or selecting existing objects from drop-down menus.

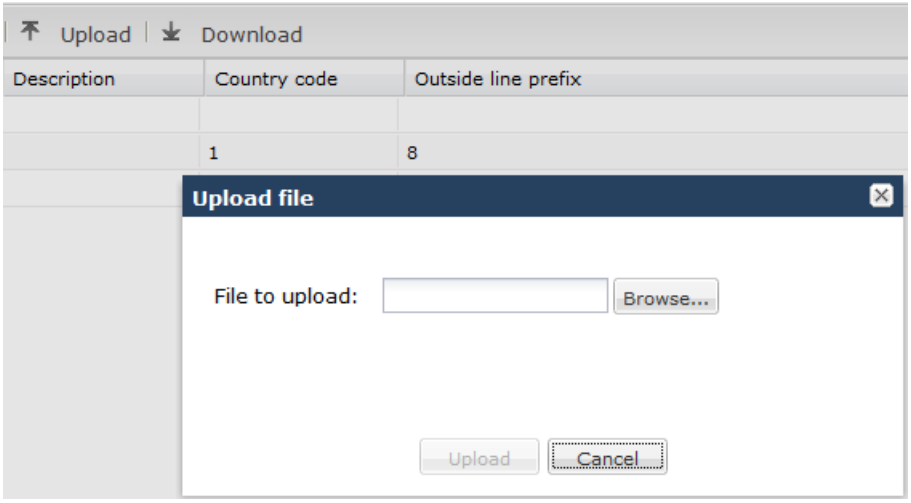
Uploading and Downloading Key Files

The Oracle Enterprise Communications Broker allows you to upload and download files containing key elements of your configuration that are more easily managed separately from the overall system configuration. This information includes:

- Dial Plans
- Dial Patterns
- User Database Entries
- Route Database Entries
- Header Manipulation Rules

> **Note:**
>
> You upload and/or download the files described in this section separately from the files managed from the System tab.

The configuration dialogs for the list above includes buttons from which you invoke uploads and downloads.



The procedure for executing upload/download is trivial. For example, clicking the Upload button produces an Upload file dialog, which includes a Browse button. When you click Browse, the system opens a browse dialog, from which you can select the correct file and upload. Download is equally self-contained. The system performs transfers to the correct system directories in the correct format without requiring any user intervention.

Oracle Enterprise Communications Broker Configuration

Once you have finished making desired changes on the Web GUI, you are able to apply and use those changes on the Oracle Enterprise Communications Broker using a two-step process, as follows:

Save - The new configuration changes are transferred to the Oracle Enterprise Communications Broker, but are not yet operational.

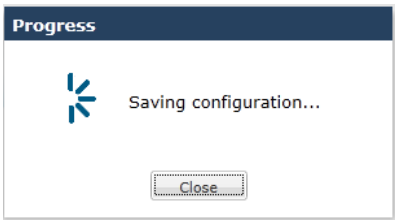
Activate - The new configuration changes are operational on the Oracle Enterprise Communications Broker.

Save and Activate

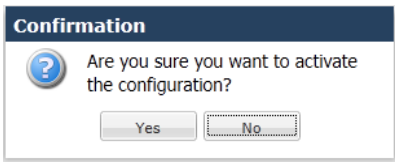
The Web GUI retains configuration changes until you send them to your device or discard them from the GUI. Configuration dialogs include an "OK" button that sends your changes to the device.

Bear in mind that you must also Save, then Activate your changes before your device actually uses your changes. The Save link, appearing as a disc icon towards the top left corner of each Web GUI page, initiates configuration Save and Activate procedures to your system.

When you click Save, the Web GUI either saves the configuration to your device or prevents you from saving invalid data. The system highlights any fields containing invalid data, allowing you to easily find and correct the mistake.



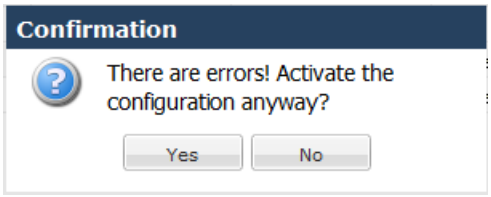
After the save is complete, the Web GUI provides you with a dialog box asking you if you wish to activate this configuration.



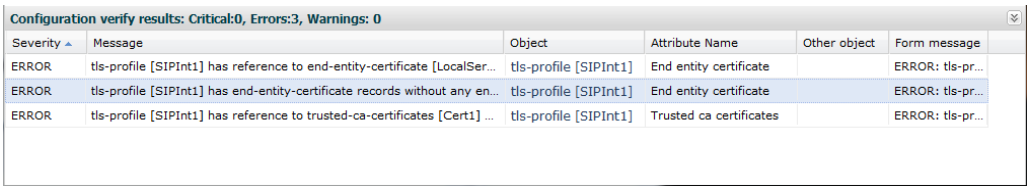
You are able to perform the save without activation, if desired. This would be common for configuration changes that need to be activated within a preferred window to avoid any service disruption.

The dialog above defaults to “No”, which leaves your changes saved to your system, but not activated. Select No if you want to activate your configuration at a later time. Select Yes to activate. The Web GUI provides a final message box indicating success when it is finished.

The Web GUI also checks your configuration for errors every time you click the Save button, indicating when it finds them prior to activation. When it discovers configuration errors, the system displays the following dialog.



The system displays configuration errors in a list at the bottom of the Web GUI. You can hide and size this error list, an example of which is displayed below. The Web GUI allows you to navigate to the each object in the list by clicking the object in the Object column.



Tool-Tips

A tool-tip is a brief description of a specific field on the configuration screens in the Web GUI. You can scroll over a field and display quick information about that field in a temporary pop-up box.

To view a tool-tip description from any configuration screen, scroll the mouse over the field title. A box displays allowing you to view a brief description about the field.

To close the tool-tip, scroll off of the field. Or click at another location within the page.

Configuration Wizards

The Wizards control, located on the Configuration tool bar, displays a list of wizards that you can use to perform system-wide configuration procedures on the Oracle Enterprise Communications Broker (ECB).

The list of wizards includes the following:

- Set boot parameters—Set the parameters required to boot the system in one dialog.
- Set entitlements—Set the maximum number of sessions to which you are licensed maximum. You can set entitlements without performing a system reboot.
- Set initial configuration—Set the parameters required for the initial configuration of the system in one dialog.
- Set license—Add licenses.
- Set login banner—Enter the text that you want users to see when they log on to the ECB.
- Set time zone—Set the time zone that you want to use for the ECB. The time zone setting does not require a system reboot.
- Upgrade software—Set the software file to upload and the upload method. The software upgrade requires a reboot to take effect. You can opt to Reboot After Upload, or you can reboot at a later time.

Using Tag Fields

The Oracle Enterprise Communications Broker provides the user with a configuration element data field referred to as a tag. The user enters information into these fields for descriptive and grouping purposes. Users establish their own criteria for labeling configuration elements with these tags. These fields have no operational effect on signaling services.

Tags are text fields for use on the following configuration objects:

- Agents
- Users
- Routes

Users can enter any text desired into the string field and can apply as many tags to a configuration object as desired. The user can then filter element list searches using tags as a means of organizing these objects. Applicable element list search fields include a down arrow that exposes a tag drop-down list, from which the user selects the tag on which they want to filter the list. Tags have no operational function other than supporting this filtering.

Home Tab

The Oracle Enterprise Communications Broker (ECB) provides a web-based dashboard on the Home tab that can display SIP data statistics to help you monitor and manage the system, for example, SIP Media Flows and Current Memory Usage. The ECB collects only SIP data for the dashboard widgets, including the default CPU and Memory widgets. For this reason, you must set up a valid SIP configuration before the ECB can display any data on a dashboard widget.

The Dashboard supports up to 18 widgets. Each widget can display up to 100 data samples in intervals of 1 hour, 1 minute, or 1 second. You can select a chart, a graph, a table, a web form,

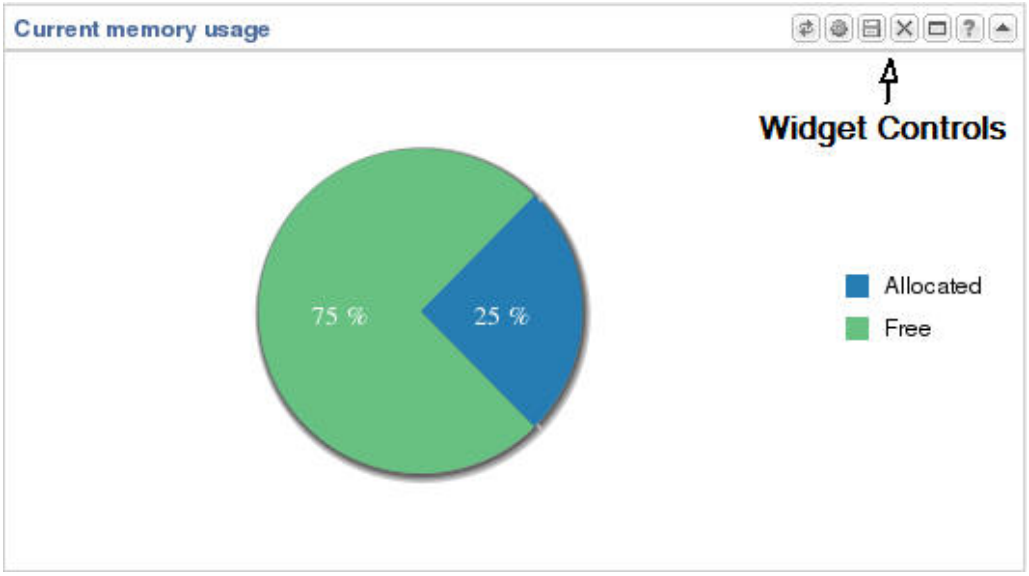
or text for the display. Customize the dashboard by adding, deleting, and moving the widgets. You can refresh the statistics displayed on the dashboard and you can reset the dashboard to its default display. The default display includes:

- Highest CPU Usage
- Current Memory Usage
- Historical Memory Usage

The following table describes the controls that you can use to customize the Home page display.

Button	Description
Refresh	Updates all of the widgets on the Dashboard.
Add widget	Displays a list of widgets that you can add to the Dashboard.
Reset	Resets the Dashboard to display the default widgets. All other widgets are removed from the Dashboard.

Use the icons in the upper right corner of the widget to perform specific tasks. Roll the mouse over the icon for a description of the function.



Note that the operation of widgets, such as those that require the SIP.Session module, may affect system performance. The system displays a warning when you add a widget that may affect performance. Oracle recommends adding such widgets at a time when the performance impact will not degrade service.

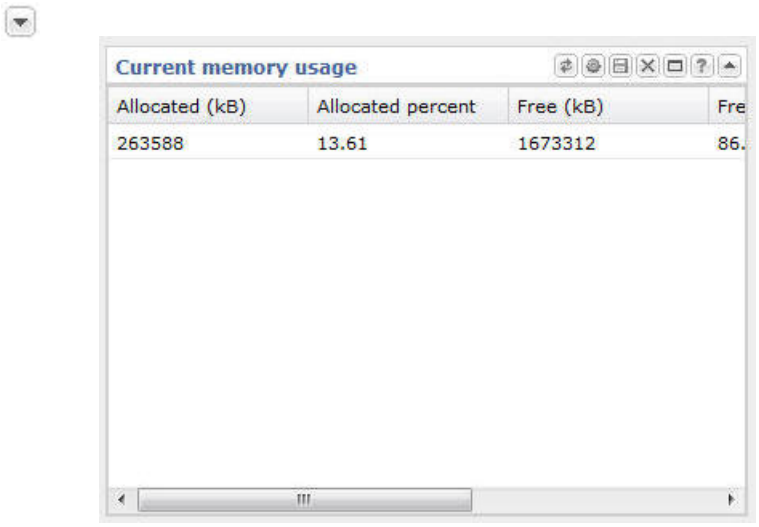
Dashboard Widgets

By default, the dashboard displays widgets applicable to the Oracle Enterprise Communications Broker.

Each widget on your Dashboard provides tools in the upper right corner of the widget that allow you to perform specific tasks. The following table provides a description of each tool in the Dashboard widget.

Tool	Description
	Refresh - Allows you to update all of the statistics that currently display in this widget.
	Settings - Allows you to configure specific settings that affect the display of the widget. Settings include: Table Name Auto-Refresh Interval **Note:** The Table Name setting is applicable to specific widgets only.
	Export - Allows you to export the data from the current widget to a .csv file. The data in the .csv file displays in table format.
	Remove - Allows you to remove the widget from the Dashboard.
	Enlarge - Allows you to enlarge the widget on the screen and place it on top.
	Help - Displays a short description of the current widget.
	Minimize - Allows you to minimize the widget in the Dashboard.

Tool	Description
	Maximize - Allows you to maximize the widget in the Dashboard.



Add a Dashboard Widget

Add a widget to the Web GUI Dashboard to display SIP and System statistics to help you monitor and manage the system.

You can add up to 18 widgets to the Dashboard with the **Add widget** control on the Web GUI Home page. The system does not require a reboot after adding a widget to the Dashboard.

> **Note:**
>
> If the system displays a warning that adding this widget requires the SIP.Message module to be enabled, the system enables the SIP.Message module when you add the widget.

1. From the Home page, click **Add widget**.
2. From the list of **Widgets**, click the name of the widget to add.
3. Under the **Command** column header, click **Add** for the widget to add.

   The system displays a success message.
4. Click **OK**.
5. Click **Close**.

   The system displays the Dashboard with the newly added widget.

See "Configure Data Sampling Settings for a Dashboard Widget."

Configure Data Sampling Settings for a Dashboard Widget

Confirm that the widget that you want to configure is on the Dashboard. See *Add a Widget*.

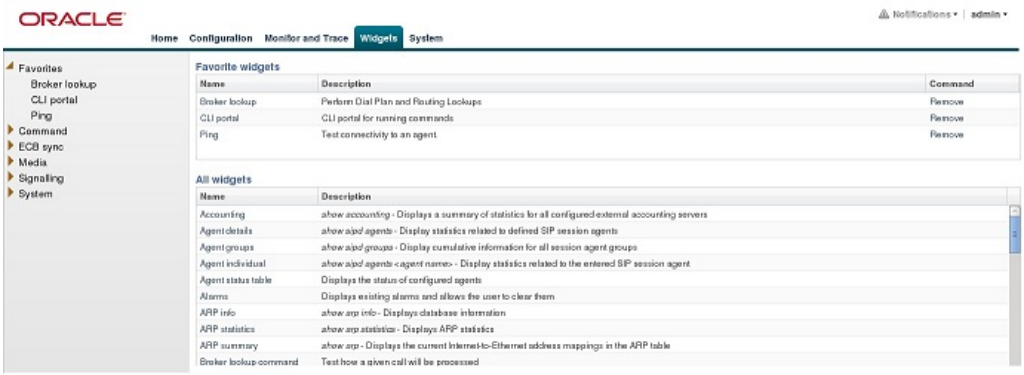
To see SIP and System statistics displayed on a Dashboard widget, the system requires a setting for how often to refresh the display. You can use the default interval or select one from the Auto-refresh interval drop-down list on the widget. Some widgets also display the Table Name drop-down list, where you can set the data sampling frequency. For example, you might configure the widget to refresh the display every 40 seconds and to display the data samples in one minute increments.

1. Click the **Home** tab.
2. On the widget, click the **Settings** icon.
3. Select a widget display refresh frequency from the **Auto-Refresh Interval (seconds)** drop down list.
4. If the widget displays the **Table Name** drop-down list, select a data sampling increment for the widget display.
5. Click **OK**.

Widgets Tab

The Widgets Tab provides a place where you can select and view graphical displays of data and statistics about the Oracle Enterprise Communications Broker (Oracle Enterprise Communications Broker) in the form of a widget. Most widgets correspond to an equivalent ACLI `show` command. A widget can display a list, a table, a graph, or text depending on the type of data and the purpose of the display.

The Widgets tab displays a categorical list of widgets in the left-hand navigation pane, and an alphabetical list of widgets in the All Widgets pane. The All Widgets list specifies the corresponding ACLI show command, where applicable, and provides a description of the data provided. You can populate the Favorite Widgets list with the widgets that you use the most often.



Some widgets display static information, while others display actionable information. For example, the IP Connections widget displays a static list of internet connections, and the Licenses widget displays a list of licenses along with controls to Add and Delete licenses. Some widgets immediately display any available information when you click the name of the widget, for example, Configuration Inventory. Other widgets require you specify a few settings before displaying information, for example, Agent Individual. Most widgets allow you to set the Auto Refresh Interval.

When you click a widget, the system displays the widget in full-screen mode with the following controls located on the top, right-hand side of the title bar.

Tool	Description
	Refresh—Update the displayed data.
	Settings—Set the display settings per widget. Settings vary per widget, and can include: • Parameters • Auto refresh interval • Table name • Exclude processes
	Export—Export the displayed data to a .csv file. The data in the .csv file displays in table format.
	Add—Add the widget to the Dashboard on the Home page.
	Add to Favorites—Add the widget to the Favorite Widgets list.
	Help—Displays a short description of the current widget.

License Widget

This release includes a widget that allows users to add, remove and view licenses. The **Widget** tab's **System** list includes this new **License Table** widget. When opened, this widget displays a list box showing each license's:

- License Name
- Session Count
- Install Date

- Begin Date
- Expire Date

Table controls include standard **Add** and **Delete** links. When the user clicks the **Add** link, the system displays The **Add License** dialog, from which the user can enter a license name and key. The **Delete** link allows the user to delete the selected license after command confirmation.

Displaying and Clearing Alarms

The Oracle Enterprise Communications Broker provides a widget that allows the user to see all current alarms that the system has triggered.

1. Click the **Widgets** tab.

   The Oracle Enterprise Communications Broker displays the widget navigation panel to the left of the associated controls.

2. Find and click the **Alarms** widget group in the **System** widget category.

   The Oracle Enterprise Communications Broker displays the **Alarms** widget display types, including the link to the **Table** display.

3. Click the **Table** link.

   The Oracle Enterprise Communications Broker displays the **Alarms** table.

4. To clear alarms, click either the **Clear** or **Clear All** links in the Alarms table's control bar.

   The **clear** link clears the selected alarm. The **Clear All** link clears them all.

Displaying Users

The Oracle Enterprise Communications Broker provides a widget that allows the user to see a list of other users currently logged into the system.

1. Click the **Widgets** tab.

   The Oracle Enterprise Communications Broker displays the widget navigation panel to the left of the associated controls.

2. Find and click the **User Management** widget group in the **System** widget category.

   The Oracle Enterprise Communications Broker displays the **Show users** widget display types, including the link to the **Table** display.

3. Click the **Table** link.

   The Oracle Enterprise Communications Broker displays the **Show users** table. Controls also include a button that allows the current user, assuming that user has administrator privileges, to disconnect another currently connected user.

Command Line Interface (CLI) Widgets

Like many devices, the Oracle Enterprise Communications Broker includes an underlying management interface called the Command Line Interface (CLI). Support technicians use this CLI to display detailed information about the system in text format. Oracle makes this information available from the graphical user interface (GUI) with CLI Widgets, available from the Oracle Enterprise Communications Broker's widget tab. These widgets can provide useful troubleshooting information as well as insight into system operation.

The CLI portal is available from the Oracle Enterprise Communications Broker's Widgets tab. The navigation sequence from the left panel's widget list is **Command** > **CLI** > **CLI portal**. When the user clicks the CLI portal link, the Oracle Enterprise Communications Broker displays the Command line portal. The portal includes **Settings** and a **Results** panel.

CLI portal settings include:

Field Name	Description
Command	A pull-down selection box allowing the user to choose the CLI command. Available commands are presented below.
Parameter	A text box allowing the user to enter additional parameter text to refine the command output with a command argument.
Auto-refresh interval	A pull-down selection box allowing the user to specify how often the system refreshes the widget's data. Available settings, in seconds, include: • never • 30 • 40 • 50 • 60

The portal includes an OK and a Cancel button. When the user clicks OK, the system adds a CLI command output window into the **Results** panel below the **Settings**. Each CLI widget provides standard widget controls on their menu bar, including a button to make the widget visible on the dashboard.

The system produces two types of CLI widgets, depending on the command invoked:

- Text Display—The system displays the output of the command in an all-text format.
- Tabular Display—The system displays the output of these commands in a table.

The user can run additional CLI commands after the first one. The CLI portal stacks widgets in the **Results** panel, minimizing previous widgets below the new widget sequentially. By maximizing widgets, the user can scroll through the **Results** panel, effectively monitoring multiple command output information simultaneously.

The portal includes a **Remove All** button that allows the user to clear all widgets from the portal at the same time.

For command and output descriptions, see the CLI Portal Reference Appendix in this document.

5

Agent Configuration

Agent configurations specify the hops by which the Oracle Enterprise Communications Broker (ECB) defines routes. Network architects select agents to identify and delineate between key logical and/or physical locations in the overall network topology. Mapping out agent topology can also identify logical or physical locations that would benefit from an agent. Agents are usually physical devices, such as proxies, SBCs, PBXs and gateways. But agents can also be logical entities, including domain names.

You can configure agents with:

- Location definition
- Context
- Translation mode
- Inbound and outbound manipulation rules
- Traffic constraints

The last two bullets are particularly impactful when the agent is the first hop in a route, but are applicable to agents multiple hops from the ECB.

Configure a Session Agent

You can enable and configure constraints that the Oracle Enterprise Communications Broker (ECB) applies to regulate session activity with the session agent.

Configure the following before you configure a session agent.

- Media profile
- Out Translation ID
- Local Response Maps
- Codec Policy
- Session Recording Server
- TLS profile
- SIP header manipulation IDs
- LDAP
- One or more target groups

The following procedure lists all of the possible attributes that you can set on a session agent. The attributes that the ECB allows you to set depends on your deployment configuration, the licenses that you own, whether or not you configured the prerequisite that a particular attribute requires, and whether or not you enabled a certain function.

1. Access the Agent configuration object.

   Configuration, **Agents**, **Agent**.

2. On the **Agents** page, do the following:

Hostname	Enter the name of the host associated with the agent in host name, FQDN, or IP address format. This field is required and the name cannot include blank spaces. The value entered here must be unique to this agent because no two agents can use the same host name. • If you enter the host name as an IP address, you do not have to enter an IP address in the optional IP address parameter. • If you enter the host name in FQDN format, and you want to specify an IP address, enter it in the optional IP address parameter.
IP Address	(Optional) Enter the IP address for the host name that you entered in FQDN format if you want to specify the IP address. Otherwise, you can leave this parameter blank to allow a DNS query to resolve the host name.
Port	Enter the number of the port associated with this agent. • 0. If you enter zero, the ECB cannot initiate communication with this agent (although it will accept calls). • 1025-65535. • The default value is 5060. If the transport method value is TCP, the ECB will initiate communication on the TCP port of the agent.
State	Select State to enable this agent.
RURI with Hostname	• Select to resolve all outgoing requests to the Session Agent to the Session Agent name in the RURI. • Deselect to resolve all outgoing requests to the Session Agent to the Session Agent IP address in the RURI. Default: deselected.
Transport protocol	Select the transport protocol for communicating with this agent. • UDP - Default • UDP+TCP • Dynamic TCP • Static TCP • Dynamic TLS • Static TLS • DTLS • TLS+DTLS • Static SCTP
TLS profile	Select a TLS profile from the drop-down list.

Description	Enter descriptive text to identify this agent.
Source context	Select the dialing context the uses when receiving a call from this agent.
Egress number translation mode	Select a number translation code from the drop-down list. The embedded Help describes the choices. Default: E164.
Number of digits for n digit dialog	Specify the number of digits to use for translation mode (n-digit dialing). Default: 4.
Prepend prefix on egress	Specify the characters to prepend to the outbound number after translation. Valid values: telephony characters 0-9, start, hash sign, and A-D. Maximum: 25 characters.
Outbound translate from number	Select to apply outbound translation to the FROM number in addition to the request-URI. Default: disabled.
Inbound header manipulation	Select a manipulation ID for the inbound header from the drop-down list.
Outbound header manipulation	Select a manipulation ID for the outbound header from the drop-down list.
Apply outbound header manipulation on	Select when you want the system to apply the outbound header manipulation. Default: next-hop-only.
Tags	Add one or more tags for organizing and associating agents.
Early media inhibit	Select to inhibit early media. Default: disabled.
Enable OPTIONS ping	Select to enable the use of OPTIONS pings to determine the status of this agent.
OPTIONS ping interval	Set the time, in seconds, for how often to ping the session agent. Range: 0-4294967295. Default: 0.
LDAP	Select the name of the ldap- group or ldap-config-group that you want this agent to use.
Additional target group	Select an additional target group from the drop-down list.
Fork group	Enter the number for the fork group number, which determines this session agent's priority in a target list. The lower the number, the higher the priority. Default: 1.
Enable REFER termination	Select to enable the REFER method for call transfer. Default: disabled.
Send NOTIFY for REFER provisional	Select when to send a NOTIFY for REFER provisional responses. • None. The system sends no intermediate NOTIFY message. • Initial. The system sends an intermediate 100 Trying NOTIFY message.

	• All. The system sends an intermediate 100 Trying NOTIFY message, plus a NOTIFY for each non-100 provisional received by the ECB.
Stop recurse	Enter one or more response codes that you want to cause this session agent to stop route recursion. Valid response code values range from 300-599. You can enter individual response codes separated by a comma, such as 301,305 or a range such as 300-380. Default: 401,407.

3. Expand **Constraints**, and do the following:

Enable constraints	Select to enable the use of constraints on this agent.
Maximum sessions	Enter the maximum number of sessions allowed for this constraint. 0-999999999.
Maximum inbound sessions	Enter the maximum number of inbound sessions allowed from this session agent. 0-999999999.
Maximum outbound sessions	Enter the maximum number of outbound sessions allowed for this constraint. 0-999999999.
Maximum burst rate	Enter the maximum number of invites allowed in a burst time period. 0-999999999.
Maximum inbound burst rate	Enter the maximum inbound burst rate in INVITEs per second from this session agent. 0-999999999.
Maximum outbound burst rate	Enter the maximum outbound burst rate in INVITEs per second from this session agent. 0-999999999.
Burst rate window size	Enter the time period, in seconds, used to measure the burst rate. 0-999999999.
Maximum sustain rate	Enter the maximum rate of session invitations allowed within the current time period for this constraint. 0-999999999.
Maximum inbound sustain rate	Enter the maximum inbound sustain rate of session invitations allowed within the current time period for this constraint. 0-999999999.
Maximum outbound sustain rate	Enter the maximum outbound sustain rate of session invitations allowed within the current time period for this constraint. 0-999999999.
Sustained window size	Enter the time period, in seconds, used to measure the sustained rate. 0-999999999.

4. Expand **Advanced**, and do the following:

SPL options	Click **Add** to add one or more SPL options.
Trunk group	Click **Add** to add one or more trunk groups used to reach this agent.
Monitoring filters	Click **Add** to add one or more monitoring filters.

5. Click **OK**.

6. Save the configuration.

Configure a Session Agent Group

Create a session agent group to define a signalling endpoint configured to apply traffic shaping attributes and information about next hops and previous hops. Session agent groups contain individual session agents in a logical grouping. Members can vary in their individual constraints.

Session agent group members do not need to reside in the same domain, network, or realm. The Oracle Enterprise Communications Broker (ECB) can allocate traffic among member session agents regardless of their location by using the allocation strategy that you select allocate traffic across the group members.

1. Access the Groups configuration object.

   Configuration, **Agents**, **Groups**.

2. On the **Groups** page, click **Add** and do the following:

Group name	Enter a name for this Session Agent Group.
Description	Enter a description of this Session Agent Group.
Selection strategy	Select a strategy for choosing this agent from the drop-down list. Default: hunt.
Stop sag recurse	Enter one or more response codes that you want to cause this session agent group to stop route recursion. Valid response code values range from 300-599. You can enter individual response codes separated by a comma, such as 301,305 or a range such as 300-380. Default: 401,407.
Try all agents	Select to enable Session Agent Group recursion. Default: disabled.
Agents	Click **Add** to add one or more Session Agents to this Session Agent Group.

3. Click **OK**.
4. Save the configuration.

Configure ENUM Servers

You can create E.164 Number to URI Mapping (ENUM) server configurations on the Oracle Enterprise Communications Broker (ECB) to resolve SIP URIs presented to the ECB in a call. An ENUM server configuration points to one or more ENUM servers from which the ECB can request resolutions. Configuring with multiple servers provides redundancy when a particular server cannot respond or provide a resolution.

- Configure a top-level domain
- Configure one or more ENUM servers to add to the ENUM settings configuration

Similar to an agent configuration, an ENUM server configuration includes number translation settings for the strings returned by the ENUM infrastructure. The ENUM configuration also includes configuration values to support the interaction with the servers.

1. Go to **Configuration**, **Agents**, **Enum server**.

2. On the Enum servers page, click **Add**, and do the following:

Name	Enter a string that uniquely identifies this ENUM configuration. You use this name in other areas of the ECB configuration to refer to this ENUM configuration. For example, in route configuration.
Top level domain	Enter the domain extension to use when querying the ENUM servers for this configuration. For example, e164.arpa. The query name is a concatenation of the number and the domain.
Servers	Enter the list of ENUM servers (an ENUM server and corresponding redundant servers) to query. Separate each server address with a space and enclose list within parentheses. **Note:** The ECB media interface does not support management traffic for ENUM. When configuring connectivity to a media interface, do not configure these resources within a media interface's subnet range. Click **Add**, and do the following: a. Enter the name of a server. b. Do one of the following: i. Click **Apply/Add another**. ii. Click **OK**.
Number translation mode	Select the translation mode required by this agent. The modes define how to format the ENUM request to accommodate the specific ENUM server. Valid values: • E164 - Default. The server can accept numbers in E164 format. • E164-no-plus. The server uses numbers in E164 format, with the exception of the plus sign. • no-country-code. The server cannot use a country code. • pattern-only. The server cannot use any string that varies from the configured pattern. • n-digit-dialing. The server requires the specified number of digits.
Number fo digits for n digit dialing	If you selected n-digit-dialing as the Number Translation Mode for this agent, specify the number of digits the ECB must send to this server.
Prepend prefix	Specify a prefix the ECB must send to this server. For example, the digit 9, which may be required to allow outbound traffic.
Advanced settings	Expand to display the following settings.

Query method	Set the strategy the ECB uses to contact ENUM servers. Valid values are: hunt—Directs all ENUM queries toward the first configured ENUM server. When the first server is unreachable, the ECB directs all ENUM queries to the next configured ENUM server, and so on. round-robin—Cycles all ENUM queries sequentially among all configured in-service ENUM servers. The ECB directs query 1 to server 1, query 2 to server 2, query 3 to server 3, and so on.
Timeout	Set the number of seconds to elapse before a query sent to a server (and its retransmissions) times out. Range: 0-4294967295 seconds.
Lookup length	Set the length of the ENUM query, starting from the most significant digit. Default: 0. Range: 1-255.
Max response size	Enter the maximum size in bytes for UDP datagrams in DNS NAPTR responses. Range: 512 (default)-65535. Oracle recommends configuring values that do not exceed 4096 bytes.
Health query number	Set a standard ENUM NAPTR query that will consistently return a positive response from the ENUM server. Blank = disabled.
Health query interval	Set the number of seconds to perpetually probe ENUM servers for health. Range: 0-65535 seconds.
ENUM options	Click **Add**, and do the following: a. Enter the name of an ENUM option. b. Do one of the following: i. Click **Apply/Add another**. ii. Click **OK**.

3. Click **OK**.
4. Save and activate the configuration.

Multi-Hop Agent Ping

The Oracle Enterprise Communications Broker's ping function can test connectivity to endpoints that are not directly adjacent to the source Oracle Enterprise Communications Broker. This multi-hop ping capability requires that the user configure special routes dedicated to sending SIP options pings to these targets.

To enable ping tests to targets that are more than one hop from the Oracle Enterprise Communications Broker, they configure routes that have the string "`ping:`" in the **Called number** field. These routes also have first agent towards the target configured in the **Route** field, and the last agent toward the target configured in the **Destination Agent** field.

The system invokes these `ping:` routes for ping traffic only. In addition, the system prioritizes SIP signaling traffic over `ping:` route traffic.

An example of a multi-hop ping route is shown in the table below.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
*	*	Target_Agnt	ping:	Adja_Agnt	0	

Having configured this route, the user can then initiate a ping to a target, or configure agent pinging to `Target_Agnt`, via `Adja_Agnt`.

The user can also set up multi-hop ping recursion by creating multiple `ping:` routes that specify the complete path to the target. To create these paths, the user can configure `ping:` routes using the `Destination Agent` and `Route` fields to define each hop in a "ping path".

Based on the two routes entries below, for example, ping attempts to reach the device defined as `Target_Agnt` follow a two-hop path:

1. Oracle Enterprise Communications Broker to `Adja_Agnt`
2. `Adja_Agnt` to `Interim_Agnt`
3. `Interim_Agnt` to `Target_Agnt`

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
*	*	Interim_Agnt	ping:	Adja_Agnt	0	
*	*	Target_Agnt	ping:	Interim_Agnt	0	

The Oracle Enterprise Communications Broker uses an agent status, as determined by OPTIONS ping, to validate all routes using that agent. Specifying an agent's status, including in-service and out-of-service, is the same for agents using either single or multi-hop ping. The system does not use routes to out-of-service agents for any signaling traffic.

6

Dial Plan Configuration

Dial plans specify how you want the Oracle Enterprise Communications Broker (ECB) to process calls and route them according to contexts and patterns that you configure. Use the Dial Plan icon on the Configuration tab to access the Dialing Contexts configuration page, where you define how you want the system to handle dial patterns.

The Dialing Contexts configuration page displays the following permanent dialing context hierarchy parents:

- Corporate—The parent of child dialing contexts of groups of users that you add, who share dialing patterns that you specify. For example, the employees in a branch office. You can add thousands of child contexts to the Corporate parent and you can set thousands of dial patterns for each child.
- Geographic—The parent of sets of child dialing contexts pre-defined by the ECB for every geographic location in the world. You can add thousands of child contexts to the Geographic parent and you can set thousands of dial patterns for each child.

Plan and configure your contexts in hierarchical priority. The ECB always uses the most specific match it can find when performing dial pattern matches. When the ECB finds no match in a child context, it searches for a dial pattern match in the parent context.

You can configure a dial plan on the ECB by way of the Web GUI or you can upload a dial plan in a .csv file.

Dial Pattern Configuration

A dial pattern defines the prefix and pattern that the Oracle Enterprise Communications Broker (Oracle Enterprise Communications Broker) receives and defines the transformation that the system performs.

Only one transformation type is valid for each prefix and pattern match. The system displays an error when you try to configure multiple transformation types. Transformation types include replacement prefix, replacement URI, and Go to context. The following table shows examples of patterns, the transformation types, and the results of the transformation.

Prefix and pattern	Transformation type	Result
8 - xxxx	Replacement prefix	Replace with configured digits, which in this case are suitable for an outside line.
911	Replacement URI	Insert the configured service URN for emergency services.
*123	Go to context	Present the prefix/pattern to another context. Matching occurs based on the target context's configured dial patterns.

Overlapping dial pattern matches in the same context that result in finding the same target number produce configuration errors. Such errors produce an ambiguity that the system cannot

resolve, so it does not forward the message. You must configure patterns, especially those that use encoding characters, very carefully to avoid ambiguities that the system cannot resolve.

When overlapping patterns result in the same target number, the system forwards to that number without error. When the system finds overlapping patterns in different contexts, the system chooses the most specific context as a match. Child contexts are more specific than parent contexts. When configuring dial patterns, Oracle recommends keeping them unique even across contexts.

Dial Pattern Encoding Characters

The Oracle Enterprise Communications Broker (ECB) allows multiple matches to a specified dial pattern, such as the ones you add to a dialing context. You can use selected characters to help you encode dial patterns to meet your needs.

In the following table, the Character column lists the encoding characters that a dial pattern allows and the Usage column explains what the characters mean in a pattern and how to use them.

Character	Usage
Brackets [ ]	Use brackets to enclose digit ranges you need to express for a pattern. TheECB parses the pattern 8[1-20]9 as 8[01-20]9, adding an implied 0 before 1. The ECB considers both values to contain the same number of digits. The ECB strictly enforces the range and the number of characters in the preceding examples, as follows: • 8019 matches • 819 does not match • 8119 matches
The "x" character	Use the x character as a wildcard in dial pattern strings. Use the "x" character can only at the end of a string. When you configure a pattern that includes an "x" character followed by digits, the system displays an error.
Parenthesis ( )	Use parenthesis to enclose wildcard characters and express a pattern. The ECB does not strictly enforce the range and the number of characters in patterns with "x" characters in parenthesis. Consider the pattern 8xx(xx): • 812 matches • 8123 matches • 81234 matches

Note that the use of encoding characters can result in overlapping dial-pattern matches. Overlapping dial-pattern matches that result in multiple targets introduce ambiguity that the ECB cannot resolve. As a result, the system does not forward the signaling.

For example, the following two dial-patterns overlap:

- 4000
- 4xxx

Double check dial patterns made up of encoding characters to avoid overlaps.

Configure a Dial Plan

You can add one or more child dialing contexts to the Corporate and Geographic parent dialing contexts to specify how you want the Oracle Enterprise Communications Broker (ECB) to extrapolate and present universal strings to the routing engine upon ingress, as well as to create target URIs for egress based on these rules.

Configuration includes specifying the applicable:

- Prefixes, including:
  - Access codes
  - Tie line digit sequences
- Services, such as 411 and 911 services
- Dialing Ranges
- Dialing Range exceptions (gaps in dialing ranges)

1. Access the Dial Plan configuration object.

   Click **Configuration**, **Dial Plan**.

2. On the Dialing Contexts page, click **Corporate** or **Geographic**, and click **Add**.

   The attributes and instructions in the next step apply to both Corporate and Geographic.

3. On the Add Dialing Context page, do the following:

Name	Enter a name for this dial plan. Required.
Geographic location	Select a geographic location from the drop-down list to use for pattern matching when the ECB cannot find a pattern match in this context's dial-patterns.
Description	(Optional) Enter a description of this dialing context.
Country code	Enter the E.164 country code for which this dialing context exists. For example, 1 for North America.
Outside line prefix	Do one of the following: • Enter the characters required for PSTN access. Valid values: 0-9, *, #, and A-D. Maximum: 25 characters. • Leave blank to allow direct dialing.
Dial patterns	1. Click **Add**, and do the following: • Remove prefix—Enter the prefix of the dial pattern entry that you want stripped for translation. Valid values: 0-9, *, #, and A-D. Maximum: 25 characters. Allow ranges in brackets []. For example, 555[2000-3999]. • Pattern—Enter telephony digits 0-9. Maximum: 25 characters. Allow ranges in brackets []. For example: 555[2000-3999]. Use x at the end of a pattern to mean 0-9. Use parens ()around optional

digits. For example: 555xx(xxxx) means 555 followed by 2 to 6 digits.

- Description—Enter a description of this dial pattern.
- Country code—Enter the country code for this dial pattern or leave blank to inherit from the context.
- Replacement prefix—Enter the replacement prefix to add to the translated number. Valid values: 0-9. Maximum: 25 characters.
- Replacement URI—Enter the URI to use without outbound translation.
- Go to context—Select the target context from the drop-down list.

2. Click **OK**.

3. (Optional) Add another dial pattern.

4. Click **OK**.
5. (Optional) Add another child dial plan.
6. Save the configuration.

7

User Configuration

User configuration is a required task. It is, in fact, the only way to assign an agent to a user. Complex VoIP deployments can derive additional benefits from the user database. Examples include contiguous dial strings being deployed across multiple PBXs, making it impossible to identify the target PBX by dial string alone.

When you need a user database, you configure entries for all source and destination numbers that you know require user database support. The user database effectively performs dial plan tasks on individual numbers. Such tasks include identifying a user's agent and source context. These entries provide the Oracle Enterprise Communications Broker with shortcuts for determining this information.

A user often represents an actual account that exists within an enterprise's dialing network. Examples of users include an extension, a subscriber or a phone number. Each user may have a source context, which translates into "Default" dialing rules for processing that user's calls using the appropriate contextual transformation rules and the agent at which they are located.

You can configure this database manually. Alternatively, you can upload user information in a format pre-configured to translate into a user database.

User-number Fields

Clicking the `Users` icon displays the user list. This list shows all Users configured on the system. The dialog includes controls to add new users, change and copy existing users, as well as upload and download pre-configured user lists to the system.

To add users, click the `Add` link. The system displays the Add User entries dialog. The table below describes the fields available from this dialog.

Field	Description
Number	Enter the E.164 number associated with this user. Do not include the + character. Ranges are supported, using the same pattern rules as in dial plans. This allows you to set, for example, an entire branch using a single entry.
Dialing context	Select or enter the name of the context that best defines this user's preferred dialing rules.
Agent	Select or enter the name of the agent to which this user is connected. This agent is the closest agent to the user.
Tags	Users enter any text desired into the Tags field and can apply as many labels to a configuration object as desired. The user can then create and filter reports using tags as a means of organizing these objects. The Tags field provides a list box with Add, Edit and Delete controls that allow configuration of individual tags.

Resolving to the Longest Match in the User Database

The Oracle Enterprise Communications Broker user database supports resolution of overlapping numbers during lookups by selecting entries that have the longest matches.

Flexibility within the means of creating user database records allows for overlapping records, which can create ambiguity when the Oracle Enterprise Communications Broker looks for a match. To resolve this ambiguity, the Oracle Enterprise Communications Broker selects the entry that matches the most digits. This flexibility imposes the requirement on the administrator to manage user database entries that take advantage of this matching as well as prevent ambiguity.

The table shows a set of overlapping user number entries in the user database. The corresponding match length for each is determined by how many digits can match exactly. Number ranges and wildcards are not part of the match length.

Pattern	Match Length
17815551111	11
17815551[000-999]	8
17815551[111-999]	8
17815551xxx	8
1781555(x)xxx	7
xxxxxxxxxxx	0

When performing a user database lookup, the system uses the entry that matches the pattern with the longest match length. The table below provides an explanation on how different dialed numbers match the numbers configured in the user database.

Dialed Number	Matching Pattern	Reason
17815551111	17815551111	This number matches all patterns, however, this is an exact match and has the highest match length (11).
17815551112	17815551[000-999] or 17815551[111-999] or 17815551xxx	This number matches all but the first pattern. Three patterns have the longest match length of 8. The selection between them is ambiguous, and therefore undefined.
17815552111	1781555(x)xxx	This number matches only the last two patterns, but this pattern has a higher match length (7).
22222222222	xxxxxxxxxxx	This number only matches the last pattern.

As shown above, if multiple entries have the same match length, the selection between them is undefined and causes the lookup to fail. This ambiguity must be resolved by careful user database entry configuration.

8

Using Policy to Refine Routing

The Oracle Enterprise Communications Broker supports policy-based routing, allowing the user to select pre-defined policies or create their own policies and apply them to routes or the Registrar. In turn, these policies impact the behavior of the applicable routes when traffic matches user-defined conditions. The user configures new policies from the Oracle Enterprise Communications Broker's **Policy** icon (or uses pre-built policies) and applies them to routes and/or the Registrar. Routes support multiple policies.

Policy provides the Oracle Enterprise Communications Broker with a generic approach to configuring routing applications. Policy configuration assumes a desired behavior that has been identified by the user. The most common objectives include:

- Establish more specific routing decisions
- Apply additional services

The Oracle Enterprise Communications Broker abstracts policy behavior into three components including:

- Route
- Condition
- Action

The combination of route and condition define when the policy applies. The action refines the way in which the traffic uses the route. Note that possible actions may include not using the route.

This generic approach to route policy provides great flexibility in policy definition, but also imposes a level of complexity on the user, requiring them to:

- Identify the application they want to create.
- Determine how to identify the applicable traffic.
  - Use or create new routes specifically for the application.
  - Define the condition that causes the system to apply the policy to the route.
- Test traffic matching and application action.
- Ensure no overlapping configurations cause the system to use or not use the policy unexpectedly.

There are two components of a policy configuration:

- Conditions:
  - codec-condition—Tell the system to determine the presence of absence of specific codecs within an offer.
  - time-condition—Specify the day(s) of the week and time(s) of the day when the system uses the policy.
- Actions:
  - routing-action—Tells the system to modify how the route is applied.

- redirect-action—Tells the system to direct traffic to the configured agent when the route and conditions match.
- outbound-translation-action—Tells the system to perform the configured outbound translation when the route and conditions match.

Policies allow for both multiple conditions and actions. All conditions must be met for the condition to be true. Alternatively, the user can configure no conditions, meaning the policy's condition is always true. When a policy includes multiple actions, it performs all of the actions in the configured order.

The user may also need to create route(s) specific to the policy. Whether an existing or newly configured route, the user configures the route to use one or more policies to complete the application configuration.

The Redirect Action

The redirect action causes the Oracle Enterprise Communications Broker (ECB) to redirect the incoming call through a particular agent by way of policy. You can use redirect to send a call to an external resource or service, such as a transcoding Session Border Controller or a call-recorder.

When applied, the policy engine performs an additional routing lookup for the call to the specified redirect agent. The system pre-pends additional hops from the redirect to the hops that were already calculated for the current route. The redirect action adjusts the routes to send the call to the specified redirect agent first, and then to the call destination.

Redirect action configuration includes the**Hairpin signaling** field. When you enable **Hairpin signaling**, the ECB routes the call to the redirect agent first, then routes it back to the ECB before sending it to the original destination. Hairpin signaling ensures that the ECB can route the call even when the redirect agent cannot reach the final destination. Note that keeping **Hairpin signaling** disabled eliminates the extra hops and extra session required when the destination is reachable by the redirect agent.

The ECB uses the same source agent, calling number, and called number parameters as the original call to reach the redirect agent. Only the dest-agent parameter gets replaced with the redirect agent specified in the redirect policy action. Take special care with default routes or routes that use a wildcard in the dest-agent field because such routes can become part of the path to the redirect agent.

Note the following details when evaluating and configuring redirect action:

- Hops incurred by the redirect action do not affect the route cost. The system determines the route-set and order-set before the redirection takes place.
- The system does not evaluate policies applied to the redirect routes, which prevents redirection loops and other undesirable behavior.
- The system uses only the first (lowest cost) redirection path, which prevents the exponential increase of backup paths.
- You can configure redirection to agent groups, which operate normally.
- The system applies the same routing parameters of the call (source agent/number and dest number) to the redirection route lookup as the original route.
- The system does not use default routes ('*' for all match patterns) for redirection. When the ECB finds no valid routes for the redirect, the ECB rejects the call.

Configuring CNAM Replacement

The Oracle Enterprise Communications Broker provides the user with the ability to specify the value of the caller name (CNAM) in the FROM header. A simple use case would consist of an enterprise inserting the name of their company into the FROM value, in place of the original caller name. The user configures the system to use other policy or routing configurations to determine when to replace a CNAM. The system applies this policy action on SIP requests immediately prior to egress.

To configure CNAM replacement, add the **cnam-masking-action** action to the desired policy. The **Modify Policy / cnam masking action** dialog includes two fields.

- **Name**—Assigns a name as an identifier to your action.
- **Display name**—Defines the text the Oracle Enterprise Communications Broker inserts as the CNAM value in the FROM header.

Note that, if no user name is in the original FROM, the Oracle Enterprise Communications Broker inserts text configured via the CNAM mask and encloses it in brackets per RFC 2822.

Using Policy to Normalize SIP Headers

The Oracle Enterprise Communications Broker supports policy-based SIP Header Normalization, allowing the user to copy and change information in headers when their user parts are Tel-URIs or SIP URIs composed of numbers. The system writes header changes caused by policy after any inbound and before any outbound manipulation performed by header manipulation rules. These policies work for registered users and targets derived from the user database or LDAP. The user configures header normalization policies from the Oracle Enterprise Communications Broker's **Policy** icon and applies them to routes and/or the Registrar. Routes support multiple policies.

The header normalization policy works with the existing outbound translation policy. Each policy consists of one or more rules by which the system changes headers to messages that match the routes or registrar to which the policy is configured. Multiple rules can be defined for the same SIP header. The system evaluates and executes rules in the configured order and changes the header values with each rule that has a valid **New value** field.

Header normalization configuration fields include:

- **SIP header name**—The name of the SIP header to be normalized. This field cannot be empty.
- **Dialing context**—Name of dialing context that defines the dialing rules to be applied to the phone number in this SIP header. An empty value in this field indicates that no dialing rules need to be applied to the corresponding SIP header in the rule.
- **Result name**—A temporary variable name to store the result of the dialing rules on the SIP header. This name has to be unique within this policy action. The value is saved between policy rule evaluation, but is not saved after the policy is fully evaluated.
- **New value**—The **Result name** whose value will be used as the new value of the SIP header. If this field is left empty, the system does not change the value of the SIP header.

If a rule specifies a SIP header name that is absent in the SIP request and the **New value** field has a valid **Result name**, the system adds the SIP header and set it to the value of **Result name**.

ANI Masking

The Oracle Enterprise Communications Broker provides a means for ensuring that the automatic number identification (ANI) presented to a service provider be recognized as a valid screened telephone number (STN) and, therefore, is not dropped by that service provider. The user configures this function as a Header Normalization policy that rewrites the applicable header with a recognizable STN. Applicable headers are DIVERSION, P-ASSERTED-IDENTIFY or FROM, depending on the service provider's requirements.

ANI Masking Configurations

This example shows a header normalization policy that ensures that the system presents an ANI as a valid STN. This assumes the user has created a dialing context from which the system can determine an STN.

Policy

Name	Description
ANI_Mask1	North American international dialing

Condition

The default condition, which requires no configuration, is to always apply the policy.

Actions

The example below configures the action named ANI_MaskforSP1, which is of the header-normalization-action element type with two actions.

SIP header name	Dialing context	Result Name	New value
From	find_Verizon_STN	STN	
Diversion		original_diversion	STN

The results of these actions follow.

1. Use the value of the "from" header and run it through a dialing context called "find_Verizon_STN" and store the result in a variable called "stn". Since new value field is empty the value of the "from" header is unaltered.
2. Since the dialing context is empty, copy value of the "Diversion" header to the variable called "original_diversion". If the incoming SIP request did not have a "Diversion" header "original_diversion" will be set to an empty string. Lastly, copy the value of the variable "STN" into the "Diversion" header.

Routes/Registrars

Configure either routes or registrars as triggers by setting their **Policy** fields to the **ANI_Mask1** policy.

Results

This configuration provides the results presented in the table below.

SIP Message Text	Mask Function	Result SIP Message Text
To: sip: 781.630.1111@oracle.com From: sip: 781.630.2222@oracle.com	Use Dialing context titled "find_Verizon_STN" to create new value titled "STN"	To: sip: 781.630.1111@oracle.com From: sip: 781.630.2222@oracle.com Diversion: sip: 978.528.1234@oracle.com

Define a Policy

The Policy icon on the Oracle Enterprise Communications Broker (ECB) Configuration tab provides access to the policy list and configuration dialogs, where you can define and manage policies.

- Analyze traffic patterns and message content before configuring a custom policy.
- Identify the message contents and metadata that allows a policy to uniquely target the traffic.
- Configure any agents the you want to use for redirected calls.

Use the following procedure to add a new policy or modify an existing one.

1. Access the Policy object. Click **Configuration**, **Policy**.

   The ECB displays the Policy Entries page.
2. On the Policy Entries page, do one of the following:
   - Click**Add** to create a new policy.
   - Click a policy in the policy list.

   The ECB displays the **Add Policy entries** dialog.
3. Enter or edit the following information.

Parameters	Instructions
Name	Enter a name for the policy.
Description	Enter descriptive text that explains the purpose of the policy.
Conditions	Set the conditions under which the system performs the actions that you specify.

Parameters	Instructions
	For codec-condition, do the following: a. Click **Add**. b. Click **codec-condition**. c. On the Add Policy / codec condition page, do the following: i. Name—Enter a name for the codec condition. ii. Do one or both of the following: i. Contains Codecs—Click **Add**, select a codec from the drop-down list, and click either **OK** or **Apply/Add another**. ii. Missing Codecs—Click **Add**, select a codec from the drop-down list, and click either **OK** or **Apply/Add another**. d. Click **OK**. For time-condition, do the following: a. Click **Add**. b. Click **time-condition**. c. On the Add Policy / time condition page, do the following: i. Name—Enter a name for the time condition. ii. Days—Click **Add**, select a day from the drop-down list, and click either **OK** or **Apply/Add another**. iii. Start time—Set the time when you want the condition to start on the specified day. iv. End time—Set the time when you want the condition to end on the specified day. d. Click **OK**.
Actions	Set the one or more actions that you want the system to perform for each specified condition. For routing-action, do the following: a. Click **Add**. b. Click routing-action. c. On the Add policy / routing action page, do the following: i. Enter a unique name for the action. ii. Select a routing mode from the drop-down list. d. Click **OK**.

Parameters	Instructions
	For redirect-action, do the following: a. Click **Add**. b. Click redirect-action. c. On the Add policy / redirect action page, do the following: i. Name—Enter a unique name for the action. ii. Redirect to agent—Select an agent to redirect the call to before sending the call to the destination. iii. Hairpin signalling—Select to enable rerouting a call back to the ECB from a redirect agent before sending the call to the destination. d. Click **OK**. For outbound-translation-action, do the following: a. Click **Add**. b. On the Add policy / outbound translation action page, do the following: i. Name—Enter a unique name for the action. ii. Egress number translation mode—Select a mode from the drop-down list. iii. Number of digits for n digit dialing—If you selected n-digit-dialing as the number translation mode, enter the number of digits the ECB must send to the agent. Range: 0-25. iv. Prepend prefix on egress—Specify the number of digits to prepend to the outbound number after translation. Valid entires include telephony digits 0-9, star, pound, and A-D. Max: 25 digits. c. Click **OK**. For constraints-action, do the following: a. Click **Add**. b. On the Add policy / constraints action page, do the following: i. Name—Enter a unique name for the action. ii. Ignore constraints—Select to ignore call admission control restraints. Default: Disabled. c. Click **OK**.

Parameters	Instructions
	For header-normalization-action, do the following: a. Click **Add**. b. On the Add policy / header-normalization action page, do the following: i. Name—Enter a unique name for the action. ii. Header normalization rules—Click **Add**, and do the following: i. Header name—Enter the name of the SIP header to normalize with dialing rules. Required. ii. Dialing context—Select a dialing context from the drop-down list. iii. Result store—Specify the temporary variable name to store the result of the dialing rules on the SIP header. iv. New value—Enter the result name to use as the new value in the SIP header. The system does not change the value of the SIP header unless you specify a value. c. Click **OK**. For cnam-masking-action, do the following: a. Click **Add**. b. On the Add policy / cnam masking action page, do the following: i. Name—Name—Enter a unique name for the action. ii. Display name—Enter the caller name replacement value to display in the SIP From header. For example, your company name. c. Click **OK**.

4. Click **OK**.
5. Save the configuration.

You must apply the policy to the Registrar and any route that you want by way of the ECB Registrar or route configuration dialogs.

Applying a Policy to a Route

The user configures an Oracle Enterprise Communications Broker route's Policy from the **Modify Routing entry** dialog. The dialog presents a list box titled **Policy**, with controls that allow the user to **Add**, **Edit** or **Delete** policies to the route. When the user clicks **Add**, the system presents a drop-down selection box displaying the names of all currently configured

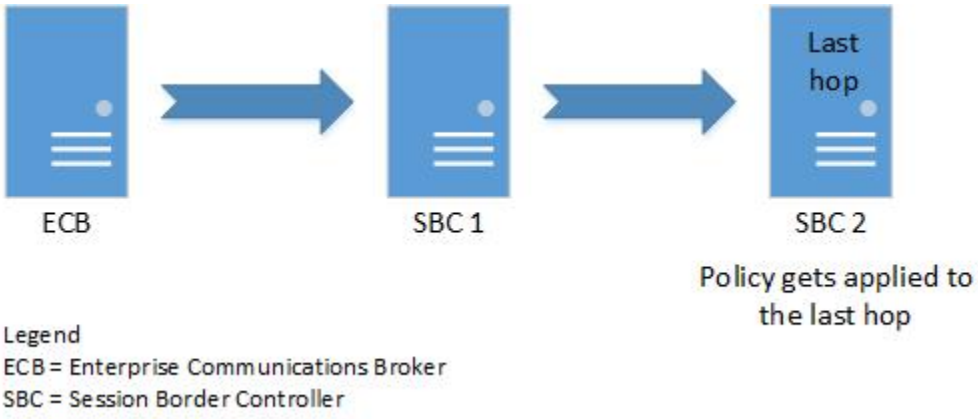
Policies. This list includes user-configured policies, as well as the system's pre-configured policies, Deny, Emergency and Stop-Recurse. The user selects the desired policy and clicks **OK**, **Add/Apply another** or **Cancel** to complete the procedure.

Runtime Routing with Policies in the User Table

You can apply a policy to the Oracle Enterprise Communications Broker (ECB) user table for more granular control over where a policy gets applied to runtime routing decisions.

Applying a policy to the user table provides more granularity than the routing table can provide because user table entries allow individual phone numbers and phone number ranges.

The ECB applies policies from the user table to the last hop on the route list specified in the routing table.



> **Note:**
>
> The system uses the policy applied to the called number from the user table.

You can use policies from the user table and the routing table together. When used together, the system obeys the policy in the user table first and the routing table second.

When you configure a policy in the User database for a particular entry, the ECB applies the policy to all agents for that entry including any Additional Target Groups among those agents.

When the session agent is a Session Agent Group, the ECB applies the policy to the session agent of that Session Agent Group.

To apply a policy to the user table:

1. Add the policy to the ECB.
2. Add the policy to a user entry in the user table.

Applying a Policy to the Registrar

The user configures the Oracle Enterprise Communications Broker Registrar with a Policy from the **Modify Registrar settings** dialog. The dialog presents a list box titled **Policy**, with controls that allow the user to **Add**, **Edit** or **Delete** policies to the registrar. When the user clicks **Add**, the system presents a drop-down selection box displaying the names of all currently configured Policies. This list includes user-configured policies, as well as the system's pre-configured policies, Deny, Emergency and Stop-Recurse. The user selects the desired policy and clicks **OK**, **Add/Apply another** or **Cancel** to complete the procedure.

Configurations Using Policy

Application examples using policy as a primary construct are presented below. These application examples are valid for both the Small and Large Enterprise models as base configurations.

Configuration examples include:

- Priority Call Handling, Using Constraint Policy Action
- Transcoding, Using Redirect Policy Action
- Call Recording, Using Redirect Policy Action
- Domestic vs. International Call, Using Translation Action
- Deny Routes, Using Routing Action
- Stop Recurse Routes, Using Routing Action
- Skip Routes, Using Routing Action

Priority Call Handling

The Oracle Enterprise Communications Broker provides support for ensuring that high priority calls, such as 911 calls, are not constrained by system utilization thresholds. Non-priority calls are still subject to system constraint configuration. The user configures priority call behavior by creating a policy with the constraint-action, then applying that policy to the applicable routes. The Oracle Enterprise Communications Broker applies this policy to the configured route, as well as every subsequent backup route, even if the backup routes are not configured with the policy.

The system ignores constraints, such as session and rate/burst limits, for calls that match the priority route. The **constraints-action** provides a built-in action called **IgnoreConstraints**. This action causes the policy to mark a call to ignore constraints. All conditions configured on the policy must be satisfied for the action to be applied. The constraints-action is special in that, when enabled, it sets the call to ignore constraints for ALL routes. This is necessary because backup routes may be utilized that are not specific to priority/emergency calls.

For example, an emergency call might first be routed to a 911 service, but on failure it would be sent to the PSTN. The call cannot be limited by constraints on the PSTN route.

The built-in **Emergency** policy is provided for user convenience, and uses the **IgnoreConstraints constraints-action**. This built-in policy can be edited, if desired.

Note that this setting does not affect the "Deny" and "StopRecurse" policies on routes. It is simply not known whether these policies are for constraining purposes or because the network infrastructure cannot support calls over those routes.

Priority Call Configurations

This example shows a policy configuration that excludes 911 calls from system constraints, ensuring that the system does not encumber the call delivery. The configuration below documents the built-in policy titled "Emergency" applied to a route for 911 calls.

Policy

Name	Description
Emergency	Built-in policy to ignore constraints for emergency or priority calls

Condition

Recall that the default condition, which requires no configuration, is to always apply the policy.

Action

Name	Enabled
IgnoreConstraints	Y

Example Route

In addition to the configurations above, the system needs this route.

Source Agent	Calling #	Destination Agent	Called #	Route	Cost	Policy
*	*	*	911	EmerSvc	0	Emergency

Transcoding and the Oracle Enterprise Communications Broker

Transcoding is the conversion of media streams' encoding between endpoints that use different codecs. The Oracle Enterprise Communications Broker allows the user to configure routing policies that identify session agents deployed for transcoding media sessions and redirect applicable signaling and media streams to those agents.

Transcoding Configurations

This example shows a policy configuration that identifies an offer that does not include PCMU or PCMA and routes the call to a Transcoding device, in this case an SBC, to ensure the media uses either PCMU or PCMA.

Policy

Name	Description
XcodePcmuPcma	Send calls that need transcoding to PCMU or PCMA to a transcoding SBC

Condition

Name	Contains Codecs	Missing Codecs
XcodePcmu		PCMU

Action

Name	Redirect to Agent	Hairpin signaling
XcodeRedirect	XcodeSBC	enabled

Example Route

In addition to the configurations above, the system includes this route.

Source Agent	Calling #	Destination Agent	Called #	Route	Cost	Policy
*	*	PBX	*	PBX	0	XcodePcmu

These configurations result in the system sending traffic that contains the PCMU codec to the PBX, and traffic without the PCMU codec to the XcodeSBC, back to the Oracle Enterprise Communications Broker and then to the PBX.

Multiple Outbound Translations

The Oracle Enterprise Communications Broker provides a means for refining outbound translation configurations using routing policy. Fixed outbound translation configuration is available at the agent level. Routing policy, however, includes an outbound translation action that takes precedence over agent and sip interface configuration and applies translation based on individual route matches.

Outbound Translation Configurations

This example shows an outbound translation configuration that configures E164 numbers for domestic calls, and E164-no-plus with 011 prepended for international calls. The example calls for two policies and two routes.

Policy

Name	Description
InternationalDialNA	North American international dialing
DomesticDialNA	North American domestic dialing

Condition

Recall that the default condition, which requires no configuration, is to always apply the policy.

Actions

Outbound Translation Action (InternationalDial)

Name	Egress number translation mode	Number of digits for n digit dialing	Prepend prefix on egress
InternationalDial	E164-no-plus	0	011

Outbound Translation Action (DomesticDial)

Name	Egress number translation mode	Number of digits for n digit dialing	Prepend prefix on egress
DomesticDial	E164	0	

Routes

In addition to the dial patterns above, the system needs two new routes.

Route#	Source Agent	Calling #	Destination Agent	Called #	Route	Cost	Policy
1	*	*	PSTN-NA	!1*	PSTN-NA	0	InternationalDialNA
2	*	*	PSTN-NA	1*	PSTN-NA	10	DomesticDialNA

Results

This configuration provides the results presented in the table below.

From	Dial String	Transformation	Result
*	15551234	15551234	Call is routed without dial string change
*	34555555	01134555555	Call is routed with dial string change

The system routes any call that includes a country code and does not begin with the digit "1" internationally, and with the applicable transformation.

Routing Action Configurations

The Oracle Enterprise Communications Broker includes three pre-configured **Policies** (based on the pre-configured **DenyAction**, **StopRecurseAction** and **IgnoreConstraints** routing actions) that the user can apply to routes. In addition, the pre-configured **Skip** routing action mode is available for easy Policy configuration, which the user can then apply to routes.

Modes for routing actions include:

- **Deny**—The Oracle Enterprise Communications Broker should not forward the call at all.

- **StopRecurse**—Stop trying backup routes if the current route fails.
- **Skip**—Do not try this route, based on condition; proceed to any backup options.

Deny Route Policy Configurations

This example shows a policy configuration that prevents the system from allowing the request's source to reach the destination. The configuration below documents the built-in policy titled "Deny" applied to an example route.

The **Deny** policy in any resultant SIP routes to a destination prevents the Oracle Enterprise Communications Broker from forwarding a SIP request to the destination point in question. The user can use the **Deny** policy to prevent sessions between two endpoints for policy or cost reasons.

The Oracle Enterprise Communications Broker includes a pre-configured deny route policy that can be used without modification.

Policy

Name	Description
Deny	Built-in policy to deny the incoming session

Condition

Recall that the default condition, which requires no configuration, is to always apply the policy.

Action

Name	Routing Mode
DenyAction	deny

Example Route

In addition to the configurations above, consider the policy applied against this route.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Class_B	*	Protect_Agnt	*	Transit_Agnt	0	Deny

The user has configured this route entry to prevent calls passing through the agent named `Class_B` from reaching any endpoint behind `Protect_Agnt`. This route entry may be installed as part of one or multiple routes that could potentially reach `Protect_Agnt`. Having assembled these routes, however, the routing engine recognizes the presence of this entry and rejects the call.

In addition, the system applies this Deny regardless of the backup route on which it is specified.

Stop Recurse Route Policy Configurations

This example shows a policy configuration that prevents the system from recursing through ensuing backup routes if the route configured with this policy fails. The configuration below documents the built-in policy titled "StopRecurse" applied to an example route.

A Stop-Recurse hop stops further attempts to forward a SIP request to a destination once the system has tried the route with the Stop-Recurse hop. The Stop-Recurse hop can be used to prevent calls from being forwarded on certain routes that are cost-prohibitive or could cause loops in SIP call flows.

The Oracle Enterprise Communications Broker includes a pre-configured stop recurse policy, with Routing Action being applied.

Policy

Name	Description
StopRecurse	Built-in policy to prevent further backup route attempts

Condition

Recall that the default condition, which requires no configuration, is to always apply the policy.

Action

Name	Routing Mode
StopRecurse	stop-recurse

Example Route

To expand upon the example, consider this route.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Class_A	*	Protect_Agnt	*	Transit_Agnt	0	StopRecurse

The user is allowing traffic to reach `Protect_Agnt` as long as it originates via the agent named `Class_A`.

For example, consider the case where an endpoint can be reached using two routes.

- Route 1: Agent_1 > Agent_2 (**StopRecurse**) > PBX
- Route 2: Agent_1 > PSTN

Route 1 has a **StopRecurse** policy defined on the hop between Agent_2 and PBX. The Oracle Enterprise Communications Broker stops processing the call if Route 1 does not receive a successful response. This configuration can prevent calls initially targeted for Route 1 from using the PSTN.

Stop Recursion by SIP Response Code

The Stop Recurse parameter provides you with control over how the system allows or stops recursion on routing by allowing you to set the specific response codes that you want the system to act upon. You can control recursion by response code globally through the SIP Interface configuration or locally through an agent or agent group configuration. Valid response code values range from 300-599. You can enter individual response codes separated by a comma, such as 301,305 or a range such as 300-380. The system uses response codes 401 and 407 for the defaults.

Skip Route Policy Configurations

This example shows a policy configuration that prevents the system from using this route based on condition. The policy contrasts with StopRecurse in that the routing engine is free to recurse through other routing options after skipping.

The Oracle Enterprise Communications Broker includes a pre-configured Skip Action, which the user can select to define their policy.

Policy

Name	Description
MySkip	Do not use this route, depending on condition.

Condition

The Time Condition Fields set below invoke the skip policy on weekdays from 9am to 5pm.

Name	Days	Start Time	End Time
When to enforce MySkip	Monday, Tuesday, Wednesday, Thursday, Friday	09:00:00	16:59:59

Action

Name	Routing Mode
SkipRoute	skip

Example Route

In addition to the configurations above, the user applies the policy against target routes.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Class_B	*	Protect_Agnt	*	Transit_Agnt	0	MySkip

The resultant configuration prevents the system from using this route to **Transit_Agnt** as a means of reaching **Protect_Agnt** on weekdays from 9 to 5.

9

Routing Configuration

The Oracle Enterprise Communications Broker performs session routing via its route configuration. Route configuration establishes hop-by-hop paths to signaling endpoints.

End stations may or may not be known by the Oracle Enterprise Communications Broker; endpoints configured within the user database are known, all others are not. Whether or not they are known by the system, the last hop (agent) leading towards that end station is often known. For this reason, the Oracle Enterprise Communications Broker builds its hop list by starting with the last agent it knows in the path and recursively adding hops (agents) needed to get to that hop. In the case where the last hop is not known, the system provides its last known hop with endpoint information and allows unknown hops to try to find the endpoint.

Oracle Enterprise Communications Broker routing configuration allows the user to specify a route's cost to specify route preference. Cost may or may not be based on monetary considerations. But the reach of an enterprise's network often does allow the user to configure routes that keep session traffic within the enterprise infrastructure rather than incurring cost associated with a service provider.

The Oracle Enterprise Communications Broker allows for a range of route preference criteria to differentiate between routing paths. Criteria includes source routing based on the agent or calling number. Target-oriented criteria is also available, allowing the enterprise to designate preferred paths for specific called numbers.

Routing Fields

Clicking the `Route` icon displays the route list. Note the Route Tree widget displayed by default at the bottom of the route list. This widget provides a graphical depiction of the route selected in the list.

To add routes, click the `Add` link. The Oracle Enterprise Communications Broker displays the Add routing entry fields.

The table below describes the fields available on this dialog.

Field	Description
Source agent	Select the agent from which the traffic must come to match the route. The default is the wildcard *, meaning to match traffic from any agent.
Calling number	Type in the number from which the traffic must come to match the route. A valid entry includes numeric characters or an FQDN resolvable via ENUM. The default is the wildcard *, meaning to match traffic with any calling number.
Dest agent	Select the agent to which the traffic must be targeted to match the route. The default is the wildcard *, meaning to match traffic to any agent.
Called number	Type in the number to which the traffic is targeted to match the route. A valid entry includes numeric characters or an FQDN resolvable via ENUM. The default is the wildcard *, meaning to match traffic with any called number.
Route	Select the agent that is the next hop in the route's path.

Field	Description
Cost	Enter a cost associated with this route to specify route use preference when there are multiple routes to the same destination A valid entry is numeric ranging from 0 to 100, with the lowest number (cost) being the preferred route.
Description	Enter any descriptive text you find helpful in identifying this route.
Tags	Users enter any text desired into the Tags field and can apply as many labels to a configuration object as desired. The user can then create and filter reports using tags as a means of organizing these objects. The Tags field provides a list box with Add, Edit and Delete controls that allow configuration of individual tags.

Route Policy

The Oracle Enterprise Communications Broker includes a route configuration control called **Policy** that allows the user to alter the behavior of routes. As the Oracle Enterprise Communications Broker assembles hops together to create an complete route, it considers and acts on the policies configured on each route entry.

The user configures a route's **Policy** from the Modify Routing entry dialog. The parameter's values include:

Policy [Deny | StopRecurse]

The field's default setting is empty, which imposes no policy on the route.

The **Deny** policy in any resultant SIP routes to a destination prevents the Oracle Enterprise Communications Broker from forwarding a SIP request to the destination point in question. The user can use the **Deny** policy to prevent session between two endpoints for policy or cost reasons. Consider this route entry.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Class_B	*	Protect_Agnt	*	Transit_Agnt	0	Deny

The user has configured this route entry to prevent calls passing through the agent named `Class_B` from reaching any endpoint behind `Protect_Agnt`. This route entry may be installed as part of one or multiple routes that could potentially reach `Protect_Agnt`. Having assembled these routes, however, the routing engine recognizes the presence of this entry and rejects the call.

To expand upon the example, consider this route.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Class_A	*	Protect_Agnt	*	Transit_Agnt	0	StopRecurse

The user is allowing traffic to reach `Protect_Agnt` as long as it originates via the agent named `Class_A`.

The **StopRecurse** policy stops further attempts to forward a SIP request to a given destination if that route fails. The **StopRecurse** policy can prevent the system from forwarding calls on routes that are cost-prohibitive or could cause loops in SIP call flows.

For example, consider the case where an endpoint can be reached using two routes.

- Route 1: Agent_1 > Agent_2 (**StopRecurse**) > PBX
- Route 2: Agent_3 > PSTN

Route 1 has a **StopRecurse** policy defined on the hop between Agent_2 and PBX. The Oracle Enterprise Communications Broker stops processing the call if Route 1 does not receive a successful response. This configuration can prevent calls initially targeted for Route 1 from using the PSTN.

Deny Patterns in Route Parameter Syntax

The Oracle Enterprise Communications Broker allows the user to configure routes that refine the routing engine's behavior, based on the called and/or calling number. Specifically, this syntax prevents the routing engine from using that route entry as a member of the applicable route sets.

The user configures this behavior by creating a route that prepends the called and/or calling numbers with an exclamation point (!). An example of a route entry using a deny pattern is shown below. The user configures the digits following the exclamation point as the first digits in the string, not the entire string.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Agent_1	!123	*	*	Agent_2	0	

As configured, traffic that includes a calling number starting with the digits 123 does not match this route. The Oracle Enterprise Communications Broker, therefore, does not use this route in any applicable route set.

In addition, the user can configure both calling and called numbers with the deny format to establish an "and" condition to the route.

Source Agent	Calling Number	Destination Agent	Called Number	Route	Cost	Policy
Agent_1	!123	*	!456	Agent_2	0	

As configured, traffic that includes a calling number starting with the digits 123 and a called number starting with the digits 456 does not match this route.

Conversely, the "and" condition allows the following traffic originating from `Agent_1` via `Agent_2` to match this route:

- From any calling number that does not start with the digits 123 to any called number.
- From any calling number that starts with the digits 123 to any called number other than one that starts with 456.
- To any called number that does not start with the digits 456.
- To any called number that starts with the digits 456 from any called number other than one that starts with 123.

Loop Sensing for PSTN Calls

When the Oracle Enterprise Communications Broker (ECB) routing table contains no explicit route to the agent, the ECB routing engine uses an implicit route that assumes the ECB is directly available to the agent. Depending on your routing needs, you might choose to configure a default route to handle calls not routed by the implicit routes. For some calls, the default route will result in a loop back to the previous hop. Such looping can cause undesirable traffic in the network and result in calls that never reach the end point. To prevent call looping, use the **next-hop-policy-condition** parameter to construct a policy for the default route that prevents the ECB from sending a call back to the previous hop.

Using the **next-hop-policy-condition** in a policy, in combination with a skip action, removes the default route as a choice for some calls and prevents the ECB from forwarding a call:

- back to the agent that sent the call.
- to a group of agents when the agent that sent the call is a member of the group.
- back to the agent when the IP address matches the host in the from-uri of the received message.
- to a group of agents when any agent in the group uses an IP address and port that matches the host in the from-uri of the received message.

The **next-hop-policy-condition** compares the next hop in the route to either the **previous hop** or the **from-uri-host**. When either **previous hop** or **from-uri-host** resolves to true, the ECB executes the **deny** action to stop call looping.

The process for configuring call loop prevention requires the following steps:

1. Set the **next-hop-policy-condition** in a policy, along with the comparison type and the deny action.
2. Add the policy to the target route.

Configure Loop Sensing for PSTN Calls

To prevent call looping, use the **next-hop-policy-condition** parameter to construct a policy for the default route that prevents the Oracle Enterprise Communications Broker (ECB) from sending a call back to the previous hop when the agent does not respond.

1. Access the Policy configuration object. **Configuration**, **Policy**.
2. On the Policy entries page, click **Add**, and do the following:

Name	Enter a name for this policy.
Description	(Optional) Enter a description of the policy.
Conditions	Click **Add**, select **next-hop-compare-condition**, and do the following: a. Enter a name for this condition. b. Select the next hop compare mode that you want from the drop-down list. c. Click **OK**.
Actions	Click **Add**, and do the following:

a. Select **routing-action** from the drop down list.
b. Enter a name for this routing action.
c. Select **deny** from the drop-down list.
d. Click **OK**.

3. Click **OK**, and do the following:
4. Save the configuration.

Apply the policy to the target route.

10

Registrar Configuration

When enabled, the Oracle Enterprise Communications Broker's registrar provides location service and registration authentication functions. The user must decide whether to use authentication and, if so, which authentication resource to utilize. Should the user decide to use a local authentication resource, they configure it via Local Subscriber Tables (LST), available from the registrar configuration. Configure external authentication resource configuration via the LDAP configuration dialogs.

Registrar Configuration Fields

To configure the Oracle Enterprise Communications Broker to act as a SIP Registrar:

1. Access the modify Registrar Settings dialog.
2. **state**—Check the checkbox to use this SIP registrar configuration element.
3. **domains**—Enter one or more domains that this configuration element will invoke SIP registration for. Wildcards are valid for this parameter. Multiple entries can be entered in quotes, separated by commas.
4. **Minimum register expiration**—Enter the expire time in seconds to be used in the REGISTER.
   - Default: 300 (5 minutes)
   - Values: Min: 0 - Max: 999999999
5. **LST file**—See the section below on the use of an LST for registration authentication.

   Note that this is your means for editing as well as creating an LST file.
6. **Credential retrieval method**—Select the method for retrieving credentials during registration authentication from the drop-down selection box. Options include None, LST and LDAP.

   If you select either LST or LDAP, the Digest realm field appears. If you select LST, the LST hash secret field also appears.
7. **Digest realm**—Enter the name (realm ID) of the host realm initiating the authentication challenge. This value defines the protected space in which the digest authentication is performed. Valid value is an alpha-numeric character string.
8. **LST hash secret**—Click the **Set** button to display the Set LST hash secret dialog. Enter and confirm the secret used in encryption and decryption of the passwords in the XML file. Once saved, this value is not echoed back to the screen in plaintext format.
9. Save and activate your changes when finished.

Local Subscriber Table

A local subscriber table (LST) is an XML formatted file that contains one or more usernames associated with a hash as encrypted or plaintext. The LST is saved locally on the Oracle Enterprise Communications Broker’s file system.

LSTs enable a standalone Oracle Enterprise Communications Broker node or high-availability (HA) pair to forego relying on an external user database. Thus the Oracle Enterprise Communications Broker does not need to communicate with a server to authenticate users. This can eliminate the operational complexity of deploying a highly available credential storage system.

LST Configuration

To configure the Oracle Enterprise Communications Broker to use LSTs for authentication, you need to create a local subscriber table configuration element that identifies that LST. The LST must include users with minimum configuration of user name and password. Alternatively, an LST entry can include an AOR and a universal number. If there is no AOR, the username is assumed to be the AOR. The universal number field assigns a universal number to all contacts registered to the AOR.

You have the option of setting the registrar to authenticate. When messages requiring authentication are received and processed by the sip registrar, the Oracle Enterprise Communications Broker uses the identified LST for authentication.

In a local subscriber table configuration, you must define an object **name**. If the filename is entered without a path, the Oracle Enterprise Communications Broker looks in the default LST directory, which is /code/lst. If the LST file is located elsewhere on the Oracle Enterprise Communications Broker, you must specify the filename and absolute path.

When the registrar configuration includes a reference to an LST, the registrar uses it as its user list. The configuration may or may not include digest authentication functionality, depending on user configuration. Additional registrar configuration includes setting the **digest realm** appropriately (this is required for authentication), and setting the hash secret. At this point you may save and activate your configuration.

Unencrypted passwords for each user in the table is computed with the MD5 hash function as follows:

```
MD5(username:digest-realm:password)
```

Configuring the Registrar with an LST

Recall that LST editing is available via the Registrar configuration. This, in fact, defines the registrar as using the LST for registration authentication as opposed to an external resource, or accepting registrations without authentication.

1. Access the **Modify Registrar settings** dialog.
2. **LST file**—Specify the LST file for this registrar. Choose an existing LST file from the drop-down box.

   See the *Troubleshooting and Maintenance* chapter's System File Management section for explanations and instructions on how to upload a pre-built LST file to your Oracle Enterprise Communications Broker. This file would become available for selection from this drop-down box. You can store multiple LST files on your system, if desired. See the LST File Format section below to learn how to build an LST file.
3. If no file exists on your system yet, you can create it here. Click the Manage LST button to display the Add local subscriber file dialog.

4. **filename**—Enter a filename for your new LST XML file. If no path is given, the Oracle Enterprise Communications Broker looks in the /code/lst directory. You may provide a complete path if the file is located elsewhere.
5. **Digest realm**—Enter the name (realm ID) of the host realm initiating the authentication challenge. This value defines the protected space in which the digest authentication is performed. Valid value is an alpha-numeric character string.
6. **Encrypt file**—Check this checkbox to cause the system to encrypt the file.
7. **Encryption secret**—Click the **Set** button to display the Set Encryption secret dialog. Enter and then confirm the secret used in encryption and decryption of the passwords in the XML file. Once saved, this value is not echoed back to the screen in plaintext format.
8. Click **OK**—This creates your LST file and allows you to add subscriber entries.

   The system displays the LST edit dialog.
9. Click the **Add** button. The system displays the Add Local Subscriber Entry dialog.

   Enter Username, Password and AoR for this subscriber. See the section on Editing an LST File for explanations on these fields.
10. **Authentication method**—Select LST from the drop-down selection box.

    The system displays the Digest realm and the LST hash secrect fields.
11. **Digest realm**— Enter the name (realm ID) of the host realm initiating the authentication challenge. This value defines the protected space in which the digest authentication is performed. Valid value is an alpha-numeric character string.
12. **LST hash secret**—Click the **Set** button to display the Set LST hash secret dialog. Enter and then confirm the secret used in encryption and decryption of the LST.
13. Save and activate your changes when finished.

Editing an LST File

Recall that LST editing is available via the Registrar configuration. This, in fact defines the registrar as using the LST. Select the LST from the drop-down list and click the Manage LST button to display the Edit Local subscriber table dialog, from which you can add, change, copy and delete users from the LST.

Additional dialog controls include:

- Verify— Parse the LST for format errors and report if the syntax/format is incorrect.
- Save—Dismiss the dialog and save all changes.
- Compare—Identify the changes made to this LST since the last save.
- Change secret—Change the hash secret used to encrypt username, digest realm and password.
- Close—Dismiss the dialog without saving changes.

Perform the following steps to add a user to the LST:

1. Click the **Add** button. The system displays the Add Local Subscriber Entry dialog.
2. **username**— Enter a username for this subscriber. Optional configuration includes, password, universal number and AoR.

The value given in the username attribute must be the same as the username that will be sent in the Authorization Header in the Request message from the users. Refer to RFC 2617 Http Authentication for details.

3. **Password**—Enter the password associated with the username of the client. This is required for all LOGIN attempts. The password displays while typing but is not saved in clear-text (i.e., *****). Valid value is an alphanumeric character string.
4. **Aor**—The Address of Record attribute is optional to specify the address of record for the subscriber if it is different than the username.
5. **Universal number**—The user's number in a format compatible for use within the routing table.
6. Repeat the subscriber add process for as many subscribers as intended.
7. Save and activate your changes when finished.

LST Runtime Execution

The LST is loaded on boot up when the configuration is appropriately set. Incoming messages thereafter can then be authenticated based on the credentials in the LST. If the Oracle Enterprise Communications Broker can not load an LST file, three things occur:

1. The following log message is recorded at the NOTICE level:

   ```
   LST [table-name] was not loaded - [filename] has error loading XML file
   ```

2. The message stated above is printed on the GUI.
3. A 503 Response is returned to the UA that sent the initial REGISTER message to the Oracle Enterprise Communications Broker.

LST Redundancy for HA Systems

The Oracle Enterprise Communications Broker synchronizes LSTs between redundant nodes to ensure that the standby node contains identical LST files. This process occurs automatically when the user uploads an LST via the GUI and when the user saves a change to an LST file via the GUI.

LST File Compression

To save local disk flash space, you can compress the LST XML file using .gz compression. The resultant file must then have an .xml.gz extension.

LST File Format

The LST file format is as follows:

```
<?xml version='1.0' standalone='yes'?>
<LocalSubscriberTable>
    <realm>aaa</realm>
    <encryption>disabled</encryption>
    <secret>02:4B:20:99:60:D2:73:4A:7B:66:B0:62:AC:8D:B5:7D:67:5F:4B:5B:47:F2:2E:
50:B5</secret>
    <subscriber username="alice" aor="alice@company.com"
universalNum="1231231234" hash="d9bfe1cac8e7fe6b79da42d03b03b96b"/>
    <subscriber username="bob" aor="bob@company.com" universalNum="1231232345"
```

```
hash="af586127536d20f4c6e88a2921780b18"/>
    <subscriber username="carol" aor="carol@vendor.com"
universalNum="1231233456" hash="b695bc18bef48e2141555e7736bd88ec"/>
</LocalSubscriberTable>
```

The LST file's elements are explained below.

localSubscriberTable

This is the head element in the XML file. Each file can have only one head element. The following attribute is found in this element:

- realm—Specifies the name of digest realm.
- encryption—This indicates whether or not the hash in the XML file is encrypted (MD5). The key for this encryption will be a preshared key and is configurable in the local subscriber table configuration element with the secret parameter.
- secret—Included if encryption is used, this is the encrypted secret.

subscriber

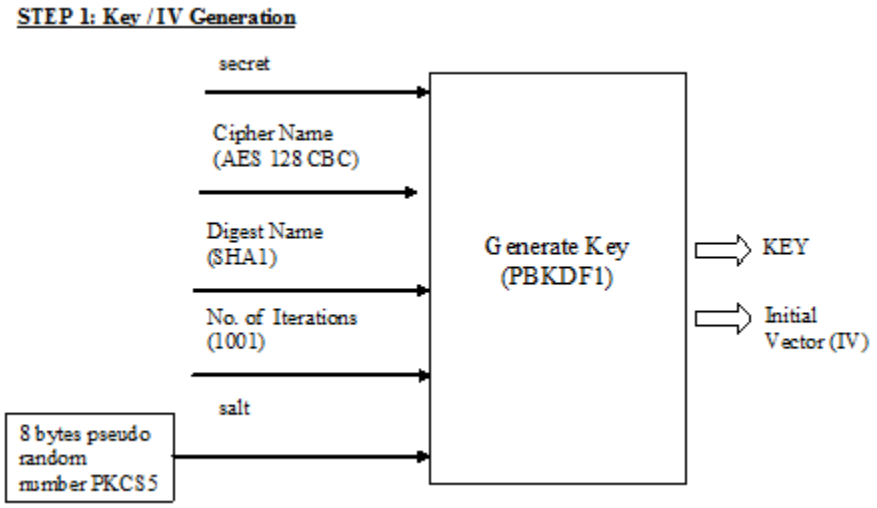
This element has the subscriber information. And has the following 5 attributes:

- username—The value given in the username attribute must be same as the username that will be sent in the Authorization header in the request message from the users. Refer RFC 2617 Http Authentication for details.
- aor—The aor attribute is optional to specify the address of record for the subscriber if it is different than the username.
- universalNum—The universalNum attribute is optional to specify the universal number (E. 164) for the subscriber.
- hash—The hash provided in the XML must be an MD5 hash of the username, digest-realm and the password of the user. This is same as the H(A1) described in RFC 2617.
  hash = md5(username:digest-realm:password)

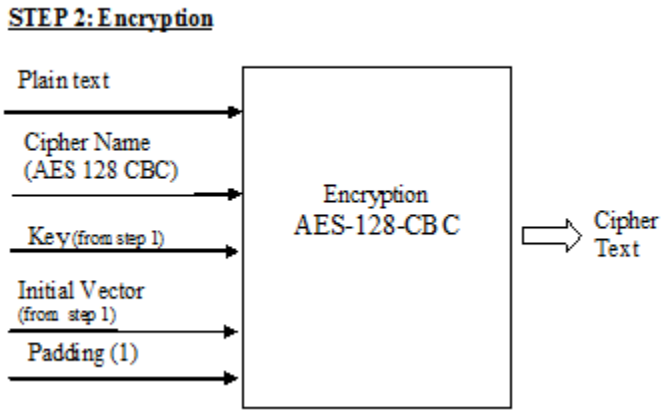
LST Subscriber Hash and Encryption

You may additionally use AES-128 CBC to encrypt the hash in the subscriber element in the LST XML file. The PSK used for encryption is configured in the **secret** parameter and an 8-byte pseudo random number is used as the salt. The LST file must set the encrypted attribute per subscriber element to true. To derive the final encrypted data you place in the XML file, three steps are performed according to the following blocks. The output of the last step, Formatting final Encrypted Data, is inserted into the LST files, subscriber element's hash value, when the encrypted attribute is set to true.

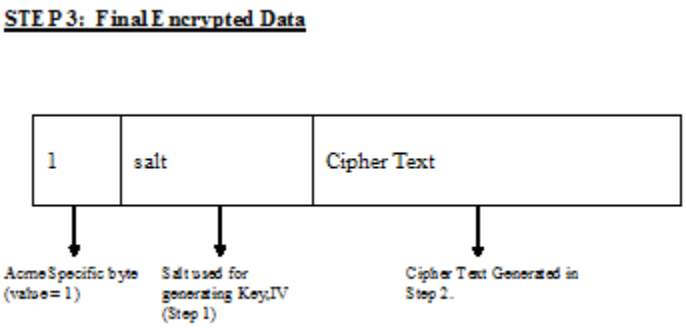
Key Initialization Vector



Encryption



Formatting Final Encryption



11

LDAP Client Configuration

The Oracle Enterprise Communications Broker supports LDAP as a communications mechanism for interaction with an LDAP server. For many enterprises, this means utilizing Active Directory, a common LDAP-based service, to request information used in SIP session routing and authentication. The Oracle Enterprise Communications Broker's LDAP client requires configuration on the Oracle Enterprise Communications Broker and the LDAP server.

Configuration aspects of LDAP client configuration include:

- LDAP server access—The user specifies LDAP server location and access preferences.
- Routing queries—The user specifies the conditions wherein the Oracle Enterprise Communications Broker performs an LDAP dip to obtain location information (home agent) for FROM and REQUEST-URIs.
- AoR queries—Optionally searches for additional AoR matches in Active Directory so that it can create additional routes to target users that have contacts stored in separate records.
- SIP Authentication queries—As an optional registration authentication mechanism, LDAP client configuration can utilize domain authentication or customized authentication server configuration on the LDAP server, as follows:
  - The use of domain authentication requires an application be installed on the domain controller.
  - Customized authentication requires the specification of compatible authentication fields on both client and server.

> **Note:**
>
> The user must ensure that phone numbers in the LDAP database are unique. If the Oracle Enterprise Communications Broker encounters multiple records with the same number, the lookup fails.

LDAP Configuration Options

The Oracle Enterprise Communications Broker provides options for LDAP configuration. These options provide the user with the flexibility to implement lookup precedence and preference on a per-agent and global basis.

An overall LDAP configuration on the Oracle Enterprise Communications Broker can include the following types of configurations.

- LDAP Config—A configuration that reaches one or multiple LDAP servers that refer to the same search base, use the same credentials, and typically service the same domain. The user can apply these configuration to agents.
- Global Config—A special LDAP configuration that the system uses as a default LDAP Config. When enabled, the system uses this LDAP Config for agents that are not

configured with an LDAP Config or LDAP Group. The user cannot rename or delete the Global Config.

- LDAP Group—Multiple LDAP configs grouped together that allow the user to refine LDAP lookups across disparate LDAP domains.

The Oracle Enterprise Communications Broker's operational precedence for using your LDAP configuration on a per call basis is as follows:

1. If the applicable agent has an LDAP group applied, calls from that agent use the group configuration.
2. If the applicable agent does not have an LDAP group configuration, but does have an LDAP Config applied, calls from that agent use the LDAP Config.
3. If the call arrives on an agent without any LDAP configuration, calls from that agent use the Global LDAP configuration.

The Global LDAP configuration is disabled by default. This, in conjunction with the absence of any LDAP Configs, excludes the use of LDAP.

Making Settings

The Oracle Enterprise Communications Broker provides access to configuration fields via lists on the GUI.

When the user clicks the LDAP icon, the system displays a navigation pane on the left side of the screen, which includes the following links:

- LDAP
- Groups

When the LDAP link is selected, they system displays the list of currently configured LDAP Configs, including the Global Config. See the *Oracle® Enterprise Communications Broker User's Guide*, Release P-CZ2.0.0 for instructions on configuring LDAP Config fields.

Clicking the Groups link displays the LDAP Groups list, from which the user configures LDAP Groups.

Configure LDAP Server Access Fields

Use the following procedure to configure the Oracle Enterprise Communications Broker (ECB) to access one or more LDAP servers.

1. Access the Modify LDAP config dialog by clicking the LDAP icon.
2. **State**—Select to enable the LDAP configuration.
3. **LDAP Servers**—Enter one or more IP addresses and optionally the port numbers for each LDAP Server that you want to add to the LDAP configuration. The first server listed is considered the primary LDAP Server, and the remaining servers are considered the secondary LDAP Servers. The HUNT strategy is used to determine the active LDAP Server (where the ECB selects the first LDAP Server; if unreachable, it selects the second LDAP Server; if that is unreachable, it selects the third LDAP Server, etc). Default ports used are 389 (for LDAP over TCP) and 636 (LDAP over TLS). IP Address must be entered in dotted decimal format (0.0.0.0). Default is blank.

4. **Username**—Enter the username that the LDAP bind request uses for authentication before access is granted to the LDAP Server. Valid values are alpha-numeric characters. Default is blank.
5. **Password**—Enter the password to pair with the username attribute, that the LDAP bind request uses for authentication before access is granted to the LDAP Server. Valid values are alpha-numeric characters. Default is blank.
6. **Ldap search base**—Enter the base Directory Number you can use for LDAP search requests. Valid values are alpha-numeric characters. Default is blank.
7. **Timeout limit**—Enter the maximum amount of time, in seconds, for which the ECB waits for LDAP requests from the LDAP server before timing out. When an LDAP response is not received from the LDAP server within the time specified, the request is retried again based on the max-request-timeouts parameter value. Valid values are 1 to 300 seconds. Default is 15.
8. **Max request timeouts**—Enter the maximum number of times that the LDAP Server is sent LDAP requests before the ECB determines that the server is unreachable and terminates the TCP/TLS connection. When an LDAP response is not received within the time specified for the timeout-limit parameter value, the request is retried the number of times specified for this max-request-timeouts value. Valid values are 0 to 10. Default is 3.
9. **TCP keepalive** —Specify whether or not the ECB keeps the TCP connection to the LPAD Server alive. Default is disabled. Valid values are:
   - enabled
   - disabled (default)
10. **Securty type**—Select the LDAP security type to use when theECB accesses the LDAP server. This parameter enables the use of LDAP over TLS (LDAPS). If you set a value for this parameter, you must also specify an ldap-tls-profile value. Default is none. Valid values are:
    - none (default) - No LDAP security type specified
    - LDAPS - Method of securing LDAP communication using an SSL tunnel. This is denoted in LDAP URLs. The default port for LDAP over SSL is 636.
11. **TLS profile**—Select the name of the Transport Layer Security (TLS) profile that the ECB uses when connecting to the LDAP Server. The ldap-sec-type must be set to **LDAPS** for this profile to apply. Valid values are alpha-numeric characters. Default is blank. See the Oracle ECB Administrator's Guide for instructions on how to create a TLS profile.
12. Save and activate the configuration.

LDAP Groups

LDAP groups on the Oracle Enterprise Communications Broker group **LDAP configs** together, allowing the user to refine lookups to multiple LDAP servers. The user configures **LDAP groups**, defines the matching criteria by which the system selects servers to query, and applies LDAP groups as profiles to agents.

When the system determines that it may find information it needs in the LDAP database, it checks to see if there is an **LDAP group** configured on the applicable agent. If there is no group, the system uses the LDAP configuration to control its lookups. This configuration can include a single LDAP Config configured on the agent or a Global LDAP config. If there is a group, the system:

1. Checks the matching criteria in the group to identify relevant LDAP servers, and

2. Performs lookups to relevant servers `using the order that the administrator has configured in the group.

If there is no match with any of the group's servers, the system does not perform an LDAP lookup and proceeds with the process sequence is uses to find information.

Matching Criteria in LDAP Groups

The Oracle Enterprise Communications Broker uses user-configured **Matching Criteria** to determine whether it should perform a lookup to each server in an **LDAP Group**. The Oracle Enterprise Communications Broker supports regex expressions within matching criteria configuration.

The system evaluates matching criteria for all LDAP servers listed in the group. If there are no matches, the system proceeds without querying LDAP. If there are any matches, the system initiates lookups to servers in the order listed in the group. If there is no match, the system skips those servers entirely.

Consider the example wherein the system performs a lookup for an phone number in LDAPGroup1. Recall that the **LDAP Group** selection is based on the agent configuration of the applicable end station. In this case, the LDAP1 agent configuration specifies LDAPGroup1 as its **LDAP group**.

The user has configured the first lookups within LDAPGroup1 to be directed to LDAP1 itself. If the number matches a criteria entry, the system adds that server to the lookup list. If not, the system skips to the next servers in the group, evaluating their **Matching Criteria**.

Matching criteria for LDAPGroup1 could include these entries.

Matching Criteria	LDAP Server
+1*	LDAP1
*@Div1.com	LDAP1
.*	LDAP2
+44*	LDAP3
+34555*	LDAP3

For LdapGroup1, the system queries LDAP1 only if the number starts with a +1 or has a host of Div1.com. The system queries LDAP2 in all cases based on its wildcard criteria. The system queries LDAP3 if the called number has a UK area code or has a Spain area code, then starts with 555.

Configuring LDAP Groups

The user must have configured LDAP configs before they can configure LDAP groups.

Follow the procedure below to create a new LDAP group:

1. Navigate to the LDAP group configuration dialogs using the sequence **Configuration** tab > **LDAP** icon > **Groups** link.

   The Oracle Enterprise Communications Broker displays the Modify LDAP group list, which includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links

2. Click the **Add** button to create a new custom policy.

The Oracle ECB displays the **Add LDAP group** dialog.

3. **Group name**—Type a name for this group into the field. The name must be between 1 and 128 alpha-numeric characters without spaces, and can include the underscore, comma, period and dash characters, as long as those are not the first characters in the name.

   The system makes this name available within all LDAP group drop-down selection boxes.

4. **Description**—Type a description for this monitor in the text box.

5. **State**—Check this checkbox to enable your LDAP group

6. The system includes the LDAP agents listbox along with the fields. This listbox includes standard **Add**, **Edit**, **Copy**, **Delete** and **Delete All** command links from which the user can access the LDAP agent dialog. Click the **Add** link to create a new group.

   The system displays the **LDAP Agents** dialog.

7. Select the desired LDAP configuration from the **LDAP config** dropdown selection box.

8. **Matching criteria**—Define the criteria the system must use to determine whether it should perform a lookup in the associated server for end station information. Regex is supported as a means of configuring matching strings.

9. Click OK to complete this configuration.

10. Save and activate your configuration.

Apply your LDAP group to the applicable Agent(s) from the **LDAP** dropdown list under the Agent's controls.

Routing Query Configuration Fields

To configure the Oracle Enterprise Communications Broker to query an LDAP database for the purpose of obtaining a call's routing information, per the Oracle Enterprise Communications Broker's processing sequence:

1. On the Modify ldap-config dialog, expand the **Routing** section using the down arrow to expose the **State**, **Route mode** and **Lookup queries** list.

2. **State**—Check the checkbox to enable the use of routing queries for your configured LDAP servers.

3. **Route mode**— Specify the route priority that the Oracle Enterprise Communications Broker uses in the route list. This parameter determines which routes are created, and the priority of those routes within the route list. Default is match-only. Valid values are:

   - match-only (default)— If there is an exact match between the dialed telephone number and an LDAP attribute value in the search response entry, a route is created corresponding to that LDAP attribute. If there is an exact match on multiple attributes, the ordering of LDAP attributes in the LDAP configuration determines the priority for each route. For example, an enterprise that uses the same phone number for both Lync and IPPBX phones, if the msRTCSIP-Line attribute is configured first, the corresponding next hop (Lync Server) would be used to create the first route in the route list.
   - attribute-order— The ordering of LDAP attributes in the LDAP configuration determines the priority for each route. So if the msRTCSIP-Line attribute is configured first, the corresponding next hop (Lync Server) would be used to create the first route in the route list. If there is a valid value present in the search response entry for a LDAP attribute, a route is created corresponding to that LDAP attribute.

- match-first— If there is an exact match between the dialed telephone number and an LDAP attribute value in the search response entry, the corresponding route gets the highest priority in the route list. For the rest of the routes, the ordering of LDAP attributes in the LDAP configuration determines the priority for each route. So if the msRTCSIP-Line attribute is configured first, the corresponding next hop (Lync Server) would be used to create the second highest priority route in the route list. If there is a valid value present in the search response entry for an LDAP attribute, a route is created corresponding to that LDAP attribute.

> **Note:**
>
> The LDAP attribute must have a valid value in the response; a match is not necessary for that attribute. If an entry is returned in the search response, there must be a match on at least one other attribute. For example, the dialed telephone number could be +17813284392 (IP-PBX Phone#), and the msRTCSIP-Line in the response could be +17814307069 (Lync phone#). A route is created for the Lync phone#, even though the dialed telephone number is the PBX Phone#.

4. Access the **Modify LDAP config / Routing / Lookup queries** dialog.

   This is a multi-element dialog that includes **Add**, **Edit** and **Delete** controls allowing you to manage your element list. Each element identifies a lookup number attribute and dialed pattern with which the Oracle Enterprise Communications Broker finds matches in the LDAP database and identifies contacts to which it builds routes. Multiple matches result in multiple targets to which the Oracle Enterprise Communications Broker creates routes for call forking.

5. **Lookup number attribute**—Enter the Active Directory attribute name. The default is telephoneNumber. Valid values are alpha-numeric characters. Some examples of Active Directory attribute names are:

   - ipPhone and msRTCSIP-Line for Lync phone number
   - telephoneNumber for IP PBX phone number
   - mobile for Mobile phone number

6. **Lookup number format type**—Select the expected attribute format from the drop down list. The default is None. Options include:

   - E164
   - E164-no-plus
   - no-country-code
   - None
   - pattern-only
   - regular-expression

7. **Lookup number regex pattern**—Enter the regular expression pattern used to break down the string of digits in the phone number extracted from the request URI and/or the FROM of the SIP request. The variables extracted from the phone number can be used in the attribute-value-format parameter. Valid values are alpha-numeric characters. The default regex is "^\+?1?(\d{2})(\d{3})(\d{4})$". This value assumes that the phone number is a North American phone number specified in the E.164 format. It extracts three variables from the phone number:

   - $1 is the area code

- $2 and $3 are the next 3 and 4 digits in the phone number

The system only queries for the home agent of the FROM if it has not already found it.

The setting only applies when **Lookup number format type** is set to **regular-expression**.

8. **Lookup number regex result**—Enter the format for the attribute value. These format values are extracted from the phone number using the extraction-regex parameter. The default parameter is "tel:+1$1$2$3". This value assumes that the phone number is a North American phone number specified in the E.164 format, and it recreates the phone number in E.164 format. Valid values are alpha-numeric characters.

   In addition to the E.164 format, the Oracle Enterprise Communications Broker uses other formats as well to store the phone numbers. You can customize the value specified for this parameter to enable successful queries for phone numbers in other formats.
   The setting only applies when **Lookup number format type** is set to **regular-expression**.
9. **Home agent attribute**—Enter the Active Directory attribute name for the agent field. The default is blank. Valid values are alpha-numeric characters. If created with the Oracle tools described in this document, the name would be orclAgentNameAttribute.
10. **Home agent regex pattern**—Enter the regular expression pattern used to break down the agent name. By default, this field is blank.
11. **Home agent regex result**—Enter the format of the regex result. By default, this field is blank.
12. **Default home agent**—Enter the name of the home agent to be used for routing if the query does not return one.
13. Click OK to save your routing query.
14. Save and activate your configuration.

Address of Record (AoR) Configuration Fields

To configure the Oracle Enterprise Communications Broker to query an LDAP database for the purpose of identifying additional AoRs that may apply to a call. The system identifies additional contacts from these AoRs and creates additional routes with which it can fork these calls:

1. On the Modify ldap-config dialog, expand the **Address of record** section using the down arrow to expose the applicable fields.
2. **Lookup number attribute**—Enter the Active Directory attribute name. The default is sAMAccountName, which is the standard Active Directory username attribute. Valid values are alpha-numeric characters.
3. **Lookup number format type**—Select the expected phone number format from the drop down list. The default is None. Options include:
   - E164
   - E164-no-plus
   - no-country-code
   - None
   - pattern-only
   - regular-expression

4. **Lookup number regex pattern**—Enter the regular expression pattern used to break down the string of digits in the phone number extracted from the request URI of the SIP request. The variables extracted from the phone number can be used in the attribute-value-format parameter. Valid values are alpha-numeric characters. The default regex is "^\+?1?(\d{2})(\d{3})(\d{4})$". This value assumes that the phone number is a North American phone number specified in the E.164 format. It extracts three variables from the phone number:
   - $1 is the area code
   - $2 and $3 are the next 3 and 4 digits in the phone number

   The setting only applies when **Lookup number format type** is set to **regular-expression**.
5. **Lookup number regex result**—Enter the format for the attribute value. These format values are extracted from the phone number using the extraction-regex parameter. The default parameter is "tel:+1$1$2$3". This value assumes that the phone number is a North American phone number specified in the E.164 format, and it recreates the phone number in E.164 format. Valid values are alpha-numeric characters.

   In addition to the E.164 format, the Oracle Enterprise Communications Broker uses other formats as well to store the phone numbers. You can customize the value specified for this parameter to enable successful queries for phone numbers in other formats.
   The setting only applies when **Lookup number format type** is set to **regular-expression**.
6. **Aor attribute**—Enter the Active Directory attribute name established to contain the AoR. The default is blank. Valid values are alpha-numeric characters.
7. **Aor extraction regex**—Enter the regular expression pattern used to break down the AoR. Valid values are alpha-numeric characters.
8. **Aor value format**—Enter the format of the regex result. The default parameter is blank.
9. Click OK to save your routing query.
10. Save and activate your configuration.

SIP Authentication Query Configuration Fields

To configure the Oracle Enterprise Communications Broker to specify an alternate authentication field in a remote database for the purpose of authenticating registration attempts:

1. On the Modify ldap-config dialog, expand the **sip-auth-query** section, directly under the **routing-query** section, using the down arrow to expose the applicable fields.
2. **Query Digest Username attribute**—Enter the name of the attribute where the digest username is stored in your LDAP database. The default is sAMAccountName, which is the standard Active Directory username attribute.
3. **Digest Auth attribute**—Enter the name of the attribute where the digest authentication hash is stored in your LDAP database. The default value is orclDigestPwdAttribute, which is a custom field populated by the oidpwdcn password filter.
4. Click OK to save your authentication query configuration.

Replacing the Calling Number in the FROM Header

The Oracle Enterprise Communications Broker provides for replacement of the calling number in SIP messages' FROM headers. Applicable messages include INVITEs that match the query, and all messages sent by the Oracle Enterprise Communications Broker to those calls' callees.

An example application is allowing recipient UEs to display a caller ID that would be recognized by the recipient, even during an enterprise's transition to new dialing schemes.

This calling number replacement function refers to LDAP resources as the source of the replacement calling number. The user configures a lookup query from the **Modify LDAP config** dialog to specify this source. Configured lookup queries become available in the **FROM header replacement** drop-down list, from which the user selects their query. This selection specifies and enables the replacement.

This feature piggybacks normal LDAP lookup procedures by collecting an additional value within the LDAP query request/response sequence. The Oracle Enterprise Communications Broker replaces the FROM header of the outgoing message with this value.

While processing this LDAP response for calling number, the Oracle Enterprise Communications Broker stores the result of the query and uses it to create the FROM header user parts for applicable outgoing messages. For traffic in which there is no match to the calling number, the Oracle Enterprise Communications Broker simply uses the original calling number.

The user can disable this replacement function by clearing the lookup query attribute name from the **FROM header replacement** field.

12
ECB Sync

The Oracle Enterprise Communications Broker allows you to configure multiple Oracle Enterprise Communications Brokers to interact with each other, sharing operational information. This functionality, called ECB Synchronization, functions similarly to layer three routers dynamically exchanging routing information. This results in extensible Oracle Enterprise Communications Broker deployments wherein any given Oracle Enterprise Communications Broker can use information from a peer Oracle Enterprise Communications Broker to make a routing decision, including simply forwarding to that peer so that it can perform the routing.

The protocol used for ECB sync is called Cluster Network Protocol (CNP). Oracle Enterprise Communications Brokers configured for synchronization share the following information, as described below:

- User Database
- Routes
- Dial Plan
- Agents

ECB sync shares information between Oracle Enterprise Communications Brokers configured as pairs, transmitter sending to the receiver. The receiver then updates its configuration with data from the transmitter. Categories of information exchanged includes:

- Routing Information - how to reach a destination.
- Context Information - how to handle a given endpoint.

ECB sync configurations begin with establishing peers. Peers operate as either:

- Transmitter—Provides its peers with its information.
- Receiver—Uses the information received from a transmitter to extend its operational scope.
- Both Transmitter and Receiver.

Any given Oracle Enterprise Communications Broker can peer with up to 10 other Oracle Enterprise Communications Brokers. A transmitter can provide its information to no more that 10 receivers; a receiver can obtain information from no more that 10 transmitters.

ECB Sync uses SIP SUBSCRIBE to establish relationships and exchange data. The data within this traffic is never presented in clear text, offering a layer of obfuscation. In addition, the systems use SIP Digest authentication for authentication/authorization of peers. Optionally, the user can configure TLS to secure ECB Sync traffic.

ECB operations are invoked upon configuration. As soon as ECB Sync is enabled and the configuration activated, the Oracle Enterprise Communications Broker can accept subscriptions as a transmitter. As soon as agents are added to the ECB Sync agent list and the configuration activated, the Oracle Enterprise Communications Broker begins to send SUBSCRIBEs to its Sync agents. Subscriptions, subscription refreshes and subscription termination all follow standard SIP procedures. Transmitters use NOTIFYs to send information to receivers. The process can use multiple, concurrent NOTIFY messages if the amount of information in the

NOTIFY exceeds maximum payload. The transmitter also compresses information; the receiver un-compresses it.

Detail on transmitter/receiver interaction includes:

- The transmitter only sends its own configuration data, not data learned from another Oracle Enterprise Communications Broker.
- The transmitter updates all of its receivers with full configuration information upon an activate if any applicable configuration information has changed. Changes, for example, to a network interface would not trigger an ECB Sync update.
- The receiver's subscription refresh interval is 1/2 the expires time (30 seconds). This timing is not configurable. If the transmitter does not receive any refresh within the expires time, it terminates the subscription.
- The transmitter does not send subsequent NOTIFY messages without positive confirmation of the previous NOTIFY by the receiver.
- The receiver resends unanswered and rejected SUBSCRIBE messages in 60 second intervals until it receives a response.
- The transmitter cancels all NOTIFY procedures upon receipt of an error message from the receiver. If this happens, the receiver restarts the NOTIFY process from the beginning. The transmitter makes this additional attempt to resend the configuration/user data only once if it receives such an error.
- If the receiver encounters an unrecoverable error, it discards all information previously learned from the transmitter, cancels the subscription and initiates a new subscription.
- The receiver un-subscribes from a transmitter gracefully upon activation of a configuration change that removes that transmitter from the receiver's list.

The Oracle Enterprise Communications Broker secures SUBSCRIBE and NOTIFY transactions using SIP Digest procedures. The transmitter challenges the receiver upon receipt of a subscription. Using SIP Digest, the receiver replies to the challenge with standard SIP Digest user, realm, password hash for the transmitter to verify. ECB sync configuration includes specifying the secret to be used for this password. Users must configure all receivers and all transmitters to use the same secret for this purpose.

The receiver tracks ECB sync information, differentiating between its own configuration and that of each transmitter. Grouping sync information by its source allows the receiver to discard the correct information if a transmitter becomes unreliable or invalid. In addition, the receiver uses this grouping to prioritize overlapping configuration information. Overlapping configuration objects include those using the identical key. These rules include:

- A receiver uses it's own object if it is in the running configuration.
- If multiple transmitters provide information on the same object, the receiver uses transmitter name alphabetical order to determine which information to use.

Receivers keep all objects from all sources in memory in case the source in use becomes invalid.

Note that all transmitters and receivers must be using the same CNP version to operate properly. If a receiver gets CNP data from a higher version, it returns a 489 "Bad Event" error and drops the payload.

ECB Sync and High Availability

A receiver updates its standby Oracle Enterprise Communications Broker with ECB sync data and status. If an active Oracle Enterprise Communications Broker goes down, the standby is,

therefore, available for immediate use as a receiver. Conversely, if a transmitter transitions from active to standby, the receiver assumes its subscription is still valid until it refreshes and receives a 481 error message. Upon receiving the 481, the receiver terminates the existing subscription and starts a new one with the new active. Note that the receiver retains learned configuration information despite the subscription change.

Synchronizing the Registration Cache

ECB Sync data includes the Oracle Enterprise Communications Broker's registration cache. The user can enable registration cache sync via a checkbox within the Sync config settings under the ECB Sync icon.

When enabled, the Oracle Enterprise Communications Broker presents its registration cache to all ECB Synch agents every nine minutes. Each ECB Sync agent uses this data to create a separate, ECB Sync-only registration cache table that includes contacts and the ECB from which it learned the cache entry.

When a call comes for a contact found in the ECB Sync-only registration cache, the Oracle Enterprise Communications Broker receiving the call adds a URI parameter to the request URI of the TO header and forwards the message to the Oracle Enterprise Communications Broker in the table. This URI parameter informs the target Oracle Enterprise Communications Broker that is must only use its registration cache for routing this call. This parameter appears as follows.

```
TO sip:user2@server2.com;orcl-regonly=true
```

In these cases, both Oracle Enterprise Communications Brokers forward the call. The Oracle Enterprise Communications Broker receiving the call uses all other routing sources to route the call, including LDAP, LST and UserDB. The Sync agent Oracle Enterprise Communications Broker routes the call using its registration cache, and skips all other routing sources, including LDAP, LST and UserDB.

Enable ECB Sync Operations

Enable ECB sync when you want one Oracle Enterprise Communications Broker (ECB) to share information with another ECB. For example, to share information about users, routes, dial plans, and agents. To use ECB Sync, you must enable the service and set the authentication secret. You can optionally enable the system to sync the configuration and the registration cache from one ECB to another.

The secret that you enter in the ECB Sync configuration must match the secret used by the ECB sync agents.

1. Access the ECB configuration page. **Configuration** > **ECB Sync**.
2. On the Modify Sync Configuration Settings page, do the following:

Attributes	Instructions
Enable sync	Select to enable ECB sync operations.
Secret	a. Click **Set**. b. Secret—Enter the secret that you use for authenticating with ECB peers. c. Confirm secret—Re-enter the secret. d. Click **OK**.

Attributes	Instructions
Configuration	(Optional)—Select to enable the system to sync the configuration from one ECB to another.
Registration	(Optional)—Select to enable the system to sync the registration cache from one ECB to another.

3. Click **OK**.

Add an ECB Sync Agent

Add an ECB Sync Agent

ECB Sync requires you to specify at least one sync agent for the Oracle Enterprise Communications Broker (ECB) to peer with before the system can perform sync operations. Each ECB can peer with up to ten ECB Sync agents. Each ECB Sync agent can peer with up to 10 ECBs.

- Configure the agents that you want to add as ECB Sync peers.
- Enable ECB Sync.

Use the optional step in the following procedure to add more agents to ECB Sync.

1. Access the Add Sync Agent Settings page. **Confiugration** > **ECB Sync** > **Sync Agent**.
2. On the Sync Agents page, click **Add**.
3. On the Add Sync Agent Settings page, select an agent from the drop-down list.
4. Click **OK**.
5. (Optional)—Repeat step 2 to add another sync agent.
6. Click **Back**.
7. Save and activate the configuration.

ECB Sync Monitoring

As described, the Oracle Enterprise Communications Broker keeps track of ECB sync peers' status and data.

This information is visible via the GUI, as follows:

- Peer Status—The ECB sync peer status widget displays all peer relationships that apply to the current Oracle Enterprise Communications Broker. Information displayed includes:
  - Status—Indicating whether the peer relationship is In Service or Out of Service. This status indicates reachability.
  - Transmitter and Receiver Sync State—Verifying subscription status for both peers.
  - Transmitter and Receiver Uptime—Displaying the uptime of the subscription.
- Peer Data—The routing table and user table include a column titled "Learned From" that displays the source of the listed element, including any ECB Sync peer.

13
HMR Configuration

Header manipulation rules (HMRs) use Oracle Enterprise Communications Broker-specific controls and/or REGEX to identify information in signaling messages the user wants to change. These rules get applied to Oracle Enterprise Communications Broker agents, and operate globally on all applicable signaling traffic that reaches that agent. This section provides instructions on configuring HMR on the Oracle Enterprise Communications Broker GUI interface.

See the Header Manipulation Appendix in this guide for full explanation of how HMRs work. This appendix is provided for those with no prior experience with HMRs. HMR configuration errors can adversely impact all of an agent's traffic. Be fully confident about the intent of an HMR, and review your HMR configurations carefully before activating them.

SIP Manipulation Configuration

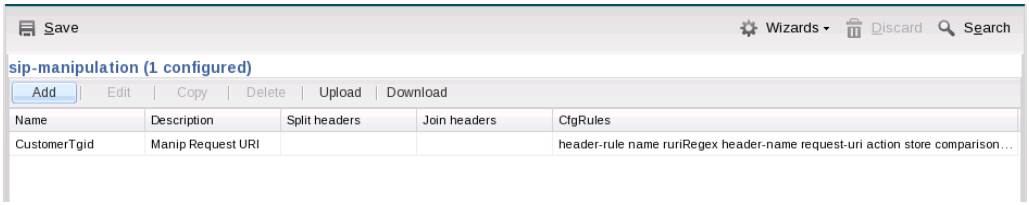
This section explains the parameters that appear in the sub-elements for the SIP manipulations configuration. Within the SIP manipulations configuration, you set up SIP header rules, and within those header rules you can configure element rules.

The appendix on Header Manipulation for HMR application includes a variety of common configuration examples. Reviewing these examples can help clarify HMR configuration and operation.

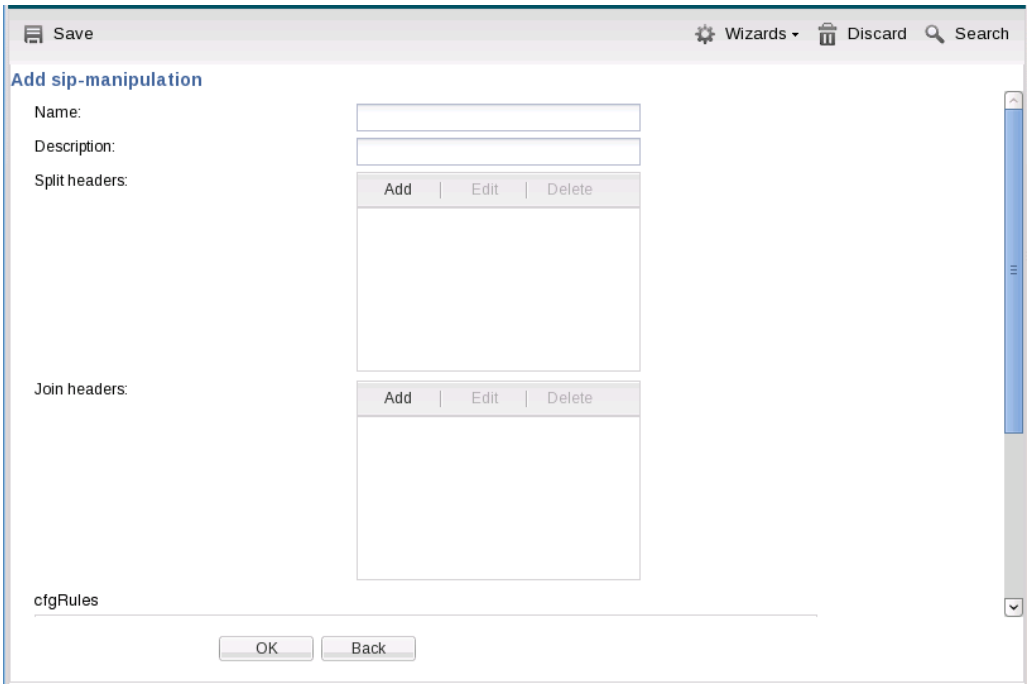
HMR Configuration Dialogs

You configure a Header Manipulation Rule (HMR) on the Oracle Enterprise Communications Broker (ECB) by way of a series of nested dialogs. You also use the HMR dialogs to assign HMRs to agents, which defines where the system uses the rule.

The first HMR dialog is the sip-manipulation list. This list shows all manually configured HMRs available on the system. The dialog includes controls to start new HMRs, change and copy existing rules, as well as upload and download pre-configured HMRs to the system.

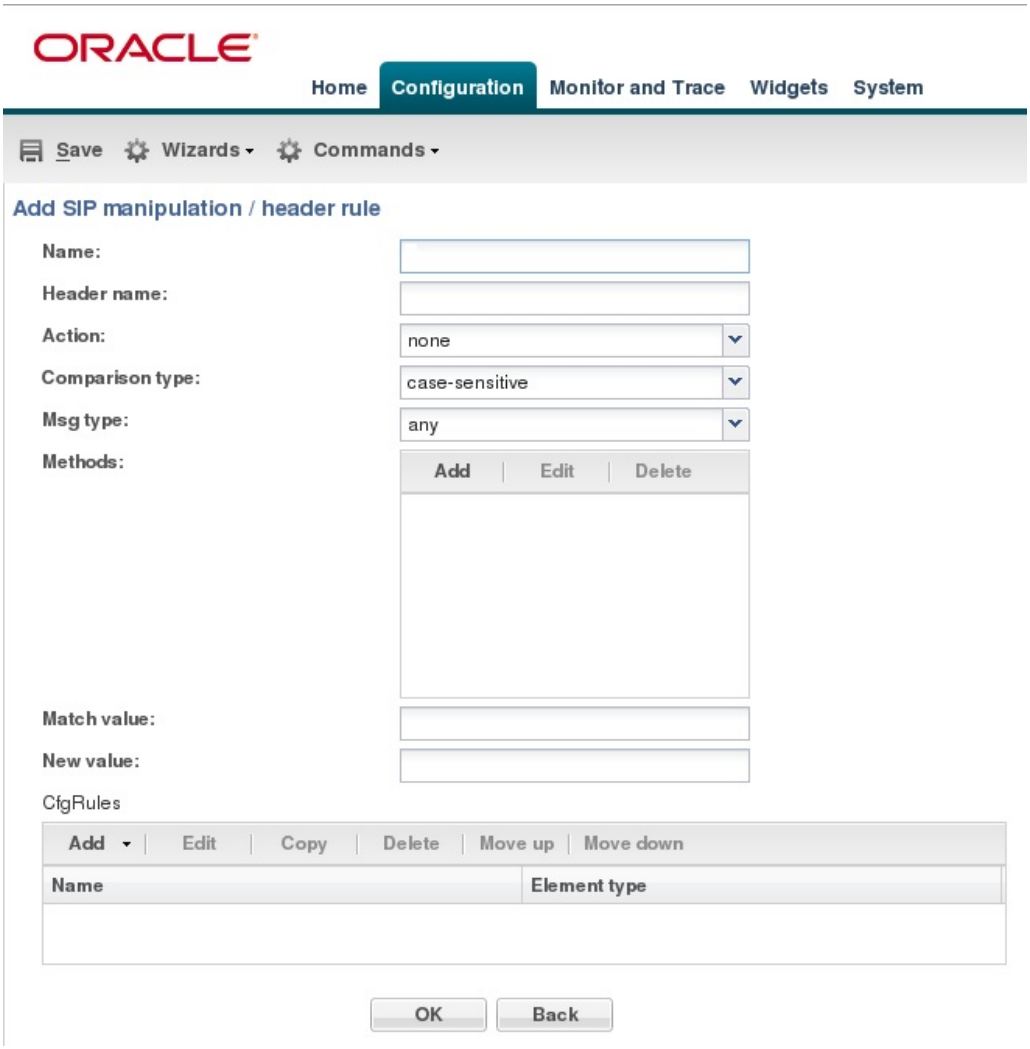


When you click the **Add**, the system displays an empty Add sip-manipulation dialog where you begin new HMR creation.



After naming and describing the HMR, you scroll to the **cfgRules** section of the dialog, which displays the list of header or MIME rules that apply to this HMR. This list includes similar controls to those of the HMR list, with the exception of the **Move** controls. Note that header rule execution order is critical when an HMR contains multiple rules. You can manage the order of HMR execution with the **Move** controls.

With the list controls, you can start a new rule or modify an existing rule. The following illustration shows the **Add sip-manipulation/header rule** dialog as an example. The header rule dialogs contain another **cfgRules** list, from which you configures element-rules, mime-header rules isup-param rules, sdp-session and sdp-media rules.



Click **OK** to accept any configuration and step back to the previous dialog. You continue configuration procedures to nest all the needed **cfgRules** under the HMR, and apply the HMR to the destination agent or location.

SIP Manipulation Fields

Descriptions of the applicable Oracle Enterprise Communications Broker configuration fields are provided below:

1. **Name**—Enter the unique identifier for this SIP Manipulation. This is the name the user selects when applying this SIP manipulation. There is no default for this value.
2. **Description**—Enter a text description for this SIP Manipulation. This is not an operational field.
3. **Split headers**—Enter a comma separated list of headers that the system separates prior to executing the manipulation. Example values are:
   - Allow,P-Asserted-Identity
   - Diversion,Allow
4. **Join headers**—Enter a comma separated list of headers that the system joins together after the manipulation execution is complete. Example values are:

- Allow,P-Asserted-Identity
- Diversion,Allow

Header Rule Fields

Descriptions of the applicable configuration fields are provided below:

1. **name**—Enter the unique identifier for this header rule. There is no default for this value.
2. **header-name**—Enter the name of the header on which you want the Oracle Enterprise Communications Broker to use this HMR. There is no default for this parameter.

   Set this parameter to @status-line, where the at-sign (@)—not allowed in SIP header names—to prevent undesired matches with header having the name status-code.
3. **msg-type**—Specify the type of message to which this SIP HMR will be applied. The default value is **any**. The valid values are:

   - any | request | reply
4. **methods**—Enter the method type to use when this SIP HMR is used, such as INVITE, ACK, or CANCEL. When you do not set the method, the Oracle Enterprise Communications Broker applies the rule across all SIP methods.
5. **comparison-type**—This choice dictates how the Oracle Enterprise Communications Broker processes the match rules against the SIP header. the default is **refer-case-sensitive.** The valid values are:

   - boolean | refer-case-sensitive | refer-case-insensitive | pattern-rule | case-sensitive | case-insensitive
6. **action**—Enter the action that you want this rule to perform on the SIP header. The default value is **none**. The valid values are:

   - add | delete | manipulate | store | none

     Remember that you should enter rules with the action type store before you enter rules with other types of actions.

     When you set the action type to store, the Oracle Enterprise Communications Broker always treats the match value you enter as a regular expression. As a default, the regular expression it uses for the match value is ,+ (which indicates a match value of at least one character), unless you set a more specific regular expression match value.
7. **match-value**—Enter the value to match against the header value in SIP packets; the Oracle Enterprise Communications Broker matches these against the entire SIP header value. This is where you can enter values to match using regular expression values. Your entries can contain Boolean operators.

   When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can use escape characters in the **match-value** parameter to support escaping Boolean and string manipulation operators.

   You can also escape the escape character itself, so that it is used as a literal string. For example, the Oracle Enterprise Communications Broker treats the string \+1234 as +1234.

   The following are escape characters: +, -, +^, -^, &, |, \, (,), ., $, ^, and “.

   You can also use the variables, $REMOTE_PORT and $LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.
8. **new-value**—When the action parameter is set to add or to manipulate, enter the new value that you want to substitute for the entire header value. This is where you can set stored

regular expression values for the Oracle Enterprise Communications Broker to use when it adds or manipulates SIP headers.

When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can use escape characters in the **new-value** parameter to support escaping Boolean and string manipulation operators.

You can also escape the escape character itself, so that it is used as a literal string. For example, the Oracle Enterprise Communications Broker treats the string \+1234 as +1234.

The following are escape characters: +, -, +^, -^, &, |, \, (,), ., $, ^, and “.

You can also use the variables, $REMOTE_PORT and $LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.

Element Rule Fields

Element rules are a subset of the SIP header manipulation rules and are applied at the element type level rather than at the entire header value.

Descriptions of the applicable configuration fields are provided below:

1. **name**—Enter the unique identifier for this element rule. There is no default for this value.
2. **parameter-name**—Enter the SIP header parameter/element on which you want the Oracle Enterprise Communications Broker to use this rule. There is no default for this parameter.
3. **type**—Specify the type of parameter to which this element rule will be applied. The default value is **none**. The valid values are:
   - header-value | header-param-name | header-param | uri-display | uri-user | uri-user-param | uri-host | uri-port | uri-param-name | uri-param | uri-header-name | uri-header

     To configure HMR so that there is impact only on the status-line; the value will be used for matching according to the **comparison-type**:
   - **status-code**—Designates the status code of the response line; accepts any string, but during the manipulation process only recognizes the range from 100 to 699.
   - **reason-phrase**—Designates the reason of the response line; accepts any string.
4. **match-val-type**—Enter the value type that you want to match when this rule is applied. The default value is **ANY**. Valid values are:
   - IP | FQDN | ANY
5. **comparison-type**—Enter the way that you want SIP headers to be compared from one of the available. This choice dictates how the Net-Net SBC processes the match rules against the SIP header parameter/element. The default is **refer-case-sensitive**.
   - boolean | refer-case-sensitive | refer-case-insensitive | pattern-rule
6. **action**—Enter the action that you want this rule to perform on the SIP header parameter/element. The default is **none**. The valid rules are:
   - add | replace | delete-element | delete-header | store | none

     Remember that you should enter rules with the action type store before you enter rules with other types of actions.

     When you set the action type to store, the Oracle Enterprise Communications Broker always treats the match value you enter as a regular expression. As a default, the regular expression is uses for the match value is .+ (which indicates a match value of at least one character), unless you set a more specific regular expression match value.

7. **match-value**—Enter the value to match against the header value in SIP packets; the Oracle Enterprise Communications Broker matches these against the value of the parameter/element. This is where you can enter values to match using regular expression values, or stored pattern matches. Your entries can contain Boolean operators.

   When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can use escape characters in the **match-value** parameter to support escaping Boolean and string manipulation operators.

   You can also escape the escape character itself, so that it is used as a literal string. For example, the Oracle Enterprise Communications Broker treats the string \+1234 as +1234.

   The following are escape characters: +, -, +^, -^, &, |, \, (,), ., $, ^, and “.

   You can also use the variables, $REMOTE_PORT and $LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.

8. **new-value**—When the action parameter is set to add or to manipulate, enter the new value that you want to substitute for the entire header value. This is where you can set stored regular expression values for the Oracle Enterprise Communications Broker to use when it adds or manipulates parameters/elements.

   When you configure HMR (using SIP manipulation rules, elements rules, etc.), you can use escape characters in the **new-value** parameter to support escaping Boolean and string manipulation operators.

   You can also escape the escape character itself, so that it is used as a literal string. For example, the Oracle Enterprise Communications Broker treats the string \+1234 as +1234.

   The following are escape characters: +, -, +^, -^, &, |, \, (,), ., $, ^, and “.

   You can also use the variables, $REMOTE_PORT and $LOCAL_PORT, which resolve respectively to the far-end and remote UDP or TCP port value.

Multi-Hop Header Manipulation Rules (HMRs)

Oracle Enterprise Communications Broker HMR support includes allowing the user to specify that a manipulation be applied depending on an agent's location (hop) in a route. Applicable hops include the next and last hop of a route. Applying an HMR when an agent is the last hop in a route is referred to as 'multi-hop' HMR. The user configures this on session agents. HMRs themselves do not require any changes to their configuration to operate as multi-hop HMRs.

The user configures an agent's **Apply-outbound-manipulation-on** parameter to specify when the system applies the agent's outbound HMR. Syntax for this command is shown below.

Apply-outbound-manipulation-on [next-hop-only | last-hop-only | next-and-last-hop]

The default setting is **next-hop-only**. This configuration makes the system apply the outbound HMR only when the agent is the next hop in the route's path.

If there are multiple HMRs the Oracle Enterprise Communications Broker must apply for the route, it applies the HMR for the last hop first. If the same agent is both next and last hop for any given traffic, the Oracle Enterprise Communications Broker applies the HMR only once regardless of the `Apply-outbound-manipulation-on` setting.

Multi-Hop Header Manipulation Rules (HMRs)

Oracle Enterprise Communications Broker HMR support includes allowing the user to specify that a manipulation be applied depending on an agent's location (hop) in a route. Applicable hops include the next and last hop of a route. Applying an HMR when an agent is the last hop

in a route is referred to as 'multi-hop' HMR. The user configures this on session agents. HMRs themselves do not require any changes to their configuration to operate as multi-hop HMRs.

The user configures an agent's **Apply-outbound-manipulation-on** parameter to specify when the system applies the agent's outbound HMR. Syntax for this command is shown below.

Apply-outbound-manipulation-on [next-hop-only | last-hop-only | next-and-last-hop]

The default setting is **next-hop-only**. This configuration makes the system apply the outbound HMR only when the agent is the next hop in the route's path.

If there are multiple HMRs the Oracle Enterprise Communications Broker must apply for the route, it applies the HMR for the last hop first. If the same agent is both next and last hop for any given traffic, the Oracle Enterprise Communications Broker applies the HMR only once regardless of the `Apply-outbound-manipulation-on` setting.

14

Monitor and Trace Tab

The Monitor and Trace tab displays the results of filtered SIP session data from the Oracle Enterprise Communications Broker. The page displays the results in a common log format for local viewing.

Monitor and Trace supports the following summary reports that you can export to a PC.

- Sessions
- Registrations
- Subscriptions
- Notable events

Each report provides sorting, searching, and paging functionality. You can customize the columns in each report and use the buttons on the page to display additional information or to perform a task.

The SIP Monitor and Trace function can store messages per session and it can store cumulative sessions across all report types. Once the sessions maximum is reached, the system removes the oldest call and adds the newest call.

- On systems with less than 4GB of RAM, the system can store:
  - 50 messages
  - 2,000 sessions
- On systems with more than 4GB of RAM, the system can store:
  - 50 messages
  - 4,000 sessions

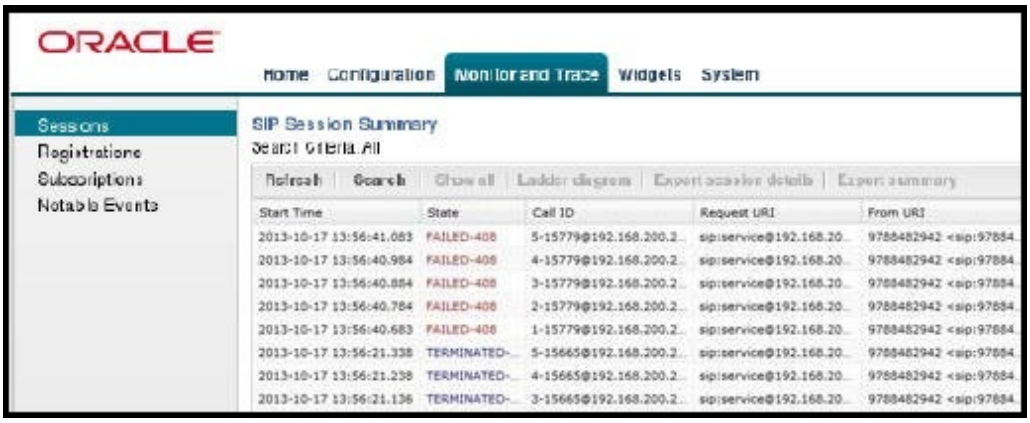
The call database is not persistent across reboots

The system can perform live paging from Monitor and Trace tables.

Sessions Report

The Sessions Report is a SIP session summary of all logged call sessions on the Oracle Enterprise Communications Broker (ECB). When Lightweight Directory Access Protocol (LDAP) is enabled on the Active Directory, LDAP session messages may also display.

The columns that display on the Sessions Report page depend on the columns that you specified in the "Customizing the Page Display" procedure.

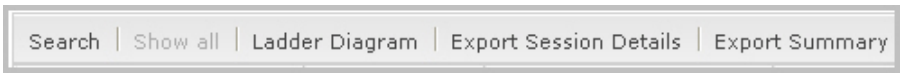


The following table describes the columns on the SIP Session Summary page.

Heading	Description
Start Time	Timestamp of the first SIP message in the call session.
State	Status of the call or media session. Valid values are: INITIAL—Session for which an INVITE or SUBSCRIBE was forwarded. EARLY—Session that received the first provisional response (1xx other than 100). ESTABLISHED—Session for which a success (2xx) response was received. TERMINATED—Session that ended by receiving or sending a BYE for an “Established” session or forwarding an error response for an “Initial” or "Early" session. The session remains in the terminated state until all the resources for the session are freed up. FAILED—Session that failed due to a 4xx or 5xx error code.
Call ID	Identification of the call source. Includes the phone number and source IP address.
Request URI	Uniform Resource Identifier (URI) formatted string that identifies a resource by way of a protocol, name, location, and any other applicable characteristic that is sent by the ECB in REQUEST headers.
From URI	URI formatted string that identifies the call source information.
To URI	URI formatted string that identifies the call destination information.
Duration	Amount of time, in seconds, that the call or media event was active.

Heading	Description
Notable Event	Indicates if a notable event has occurred on the call session. Valid values are: short session—Sessions that do not meet a minimum configurable duration threshold. Session dialogue, captured media information, and termination signalling. Any event flagged as a short session interesting event. local rejection—Sessions locally rejected at the ECB for any reason, for example, Session Agent (SA) unavailable, no route found, SIP signalling error, and so on. Session dialogue, capture media information, and termination signalling. Any event flagged as a local rejection interesting event.
Session ID	Identification assigned to the call session.
Ingress Src Addr	Source IP address of the incoming call or media event.
Egress Dest Addr	Destination IP address of the outgoing call or media event.

The following table describes the controls on the SIP Session Summary page.

Button	Description
	
Search	Use to specify parameters for performing a search for specific session summary records within the current report.
Show all	Use to display all of the session summary records in the Sessions Report.
Ladder Diagram	Use to display a Ladder Diagram of a specific record in the table. The Ladder Diagram displays detailed information about a call session or media event.
Export Session Details	Use to export the SIP messages and media events associated with the selected session to a file in text format on the local machine.
Export Summary	Use to export all logged session summary records to a file in text format on the local machine.

Display a Sessions Report

1. From the Web GUI, click **Monitor and Trace** > **Sessions**.

   The system displays the SIP Session Summary page.

2. Use the buttons on the top of the page to find, view, and export information about the records in the report.

Ladder Diagram

A ladder diagram in the Web GUI schematic that shows the call and media flow of packets on ingress and egress routes by way of the Oracle Enterprise Communications Broker.

A ladder diagram for the Sessions Report displays the following session summary information:

- Quality of Service (QoS) statistics for call sessions
- SIP messages and media events in time sequence

To display a ladder diagram for a specific record in the Sessions Report, click a record in the summary table or click **Ladder diagram** on the SIP Sessions Summary page.

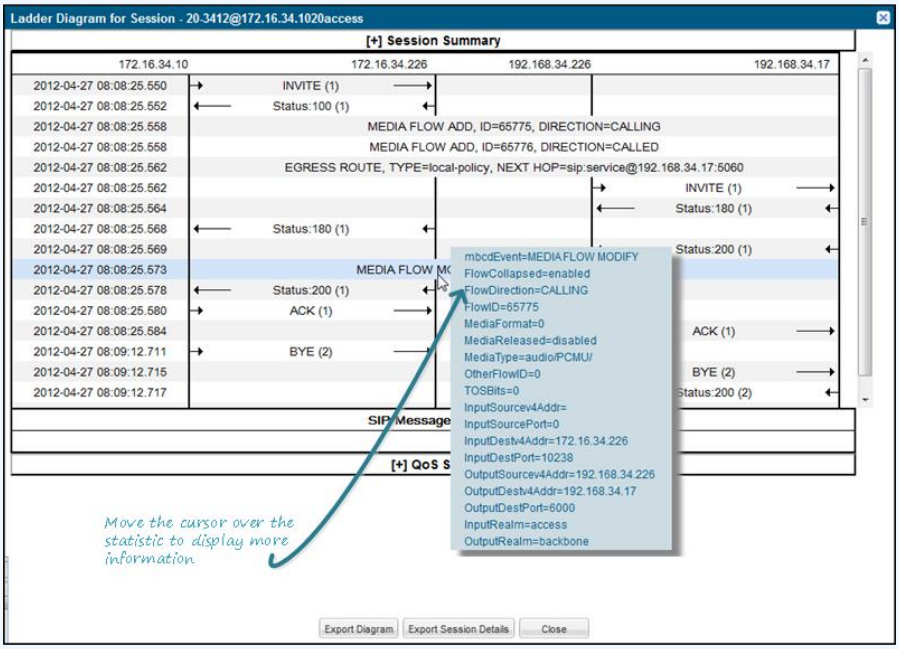
Display a Ladder Diagram

To display a ladder diagram:

1. On the Sessions Report page, click **Ladder diagram**, or select a record in the table and double-click on that record. The following is an example of the ladder diagram that displays.

Note:

The Oracle Enterprise Communications Broker (ECB) captures SIP messages, applies the Header Manipulation Rules (HMR) configured on the ECB , and then applies the Session Plug-in Language (SPL) to that message. When the message is sent out from the ECB, it applies the SPL, the HMR, and then sends out the captured SIP message. Therefore, when viewing the session detail on a Ladder Diagram, the HMR and SPL information may be present.



The Session Record Ladder Diagram consists of the following information:

- **Session Summary** - summary information about the call or media session in focus.
- **SIP Message Details** - SIP message and call flow information about the call or media session in focus.
- **QoS Statistics** - Quality of Service (QoS) statistic information about the call or media session in focus.

  You can move your mouse over any statistic in the Ladder Diagram to view additional parameters and associated values for the statistic in a pop-up window.

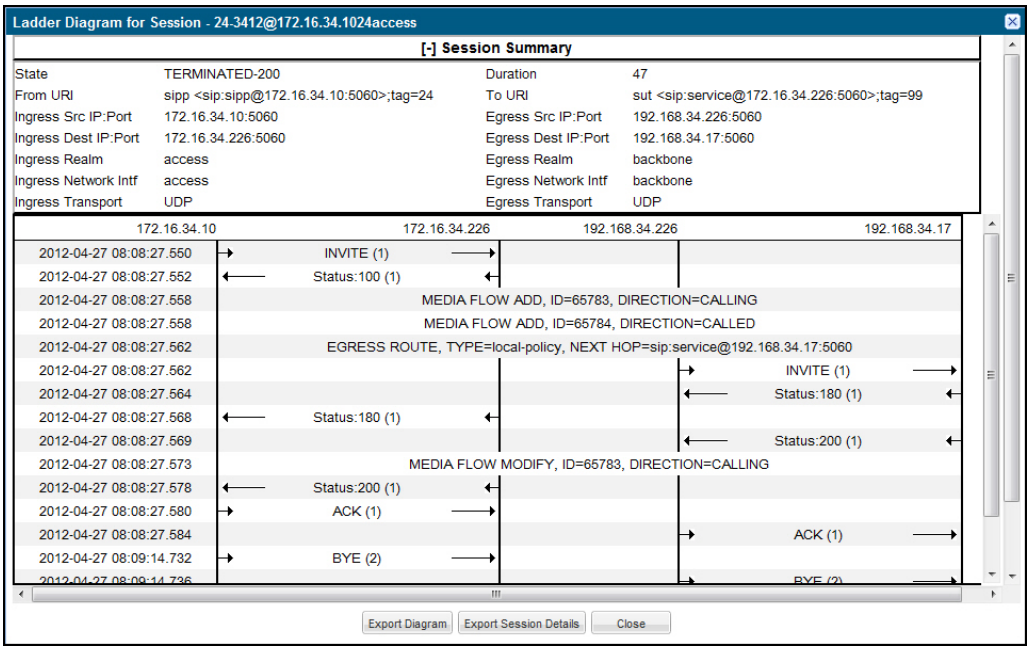
  The following table describes the buttons in this Ladder Diagram window.

Button	Description
Export Diagram	Exports all of the information in the Ladder Diagram (Session Summary, SIP Message Details, and QoS statistics), to a file in text format on the local machine.

Button	Description
Export Session Details	Exports detailed information about the SIP messages and media events associated with the session in focus, to a file in text format on the local machine.
Close	Closes the Ladder Diagram window.

Session Summary

The Session Summary window in the Ladder Diagram displays an overall summary of the call or media session in focus.



Display the Session Summary

To display the Session Summary:

1. In the Ladder Diagram, click the **[+]** next to Session Summary at the top of the Ladder Diagram window. The Session Summary window expands. This window displays a summary of information about the call or media session in focus. The following list describes each field in the Session Summary window.

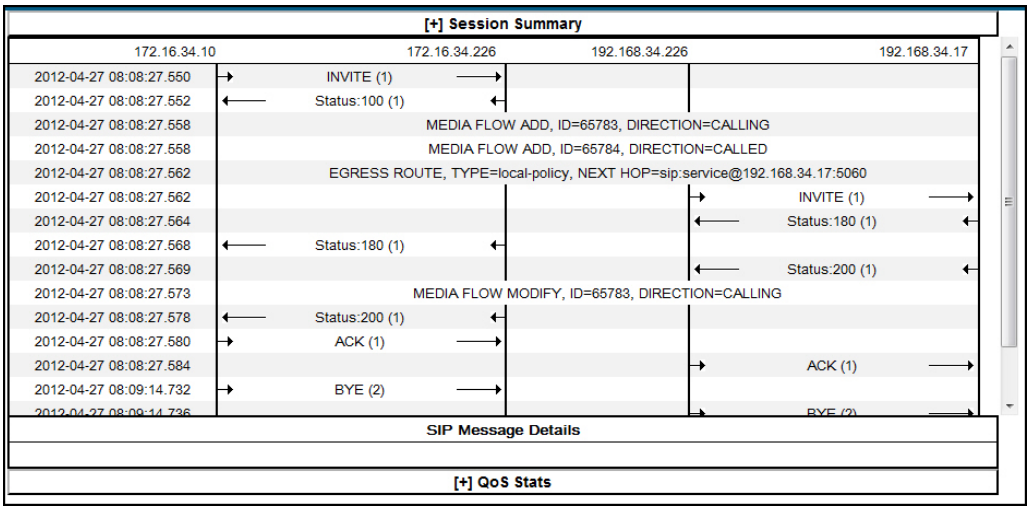
State	Status of the call or media session. Valid values are: INITIAL Session for which an INVITE or SUBSCRIBE was forwarded. EARLY Session received the first provisional response (1xx other than 100). ESTABLISHED Session for which a success (2xx) response was received.

	TERMINATED Session that has ended by receiving or sending a BYE for an "Established" session or forwarding an error response for an "Initial" or Early session. The session remains in the terminated state until all the resources for the session are freed up. FAILED Session that has failed due to a 4xx or 5xx error code.
Duration	Amount of time, in seconds, that the call or media session was active.
From URI	URI formatted string that identifies the call source information.
To URI	URI formatted string that identifies the call destination information.
Ingress Src IP:Port	Source IP address and port number of the incoming call or media session.
Egress Src IP: Port	Source IP address and port number of the outgoing call or media session.
Ingress Dest IP:Port	Destination IP address and port number of the incoming call or media session.
Egress Dest IP: Port	Destination IP address and port number of the outgoing call or media session.
Ingress Realm	Incoming realm name.
Egress Realm	Outgoing realm name.
Ingress Network Intf	Name of the incoming network interface on the Oracle Enterprise Communications Broker (ECB).
Egress Network Intf	Name of the outgoing network interface on the ECB.
Ingress Transport	Protocol type used on the incoming call or media session. Valid values are User Datagram Protocol (UDP) or Transport Control Protocol (TCP).
Egress Transport	Protocol type used on the outgoing call or media session. Valid values are User Datagram Protocol (UDP) or Transport Control Protocol (TCP).

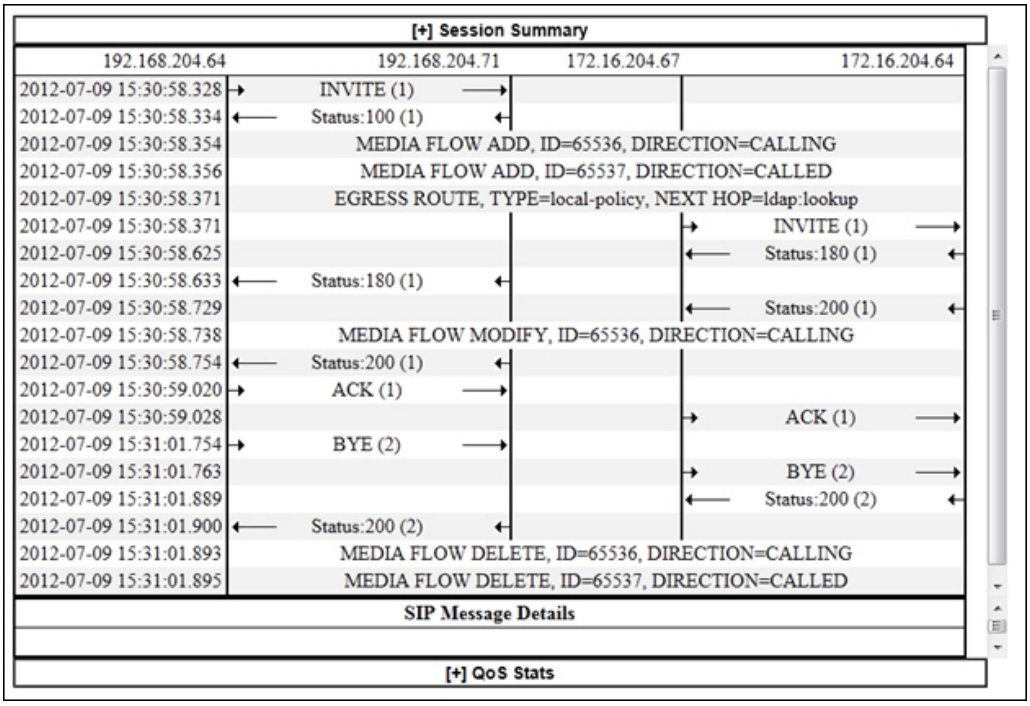
2. Click **[-]** to close the Session Summary window.

SIP Message Details

The SIP Message Detail window displays detailed information and data flow (ingress and egress) about the call or media event.



When a session is routed using the a Lightweight Directory Access Protocol (LDAP) configuration (Active Directory) for the local policy, the LDAP information displays in the Session Summary window. The next hop value containing "enum:…" or "dns:…" displays. Similarly, the next hop value "ldap:…" displays for LDAP queries.



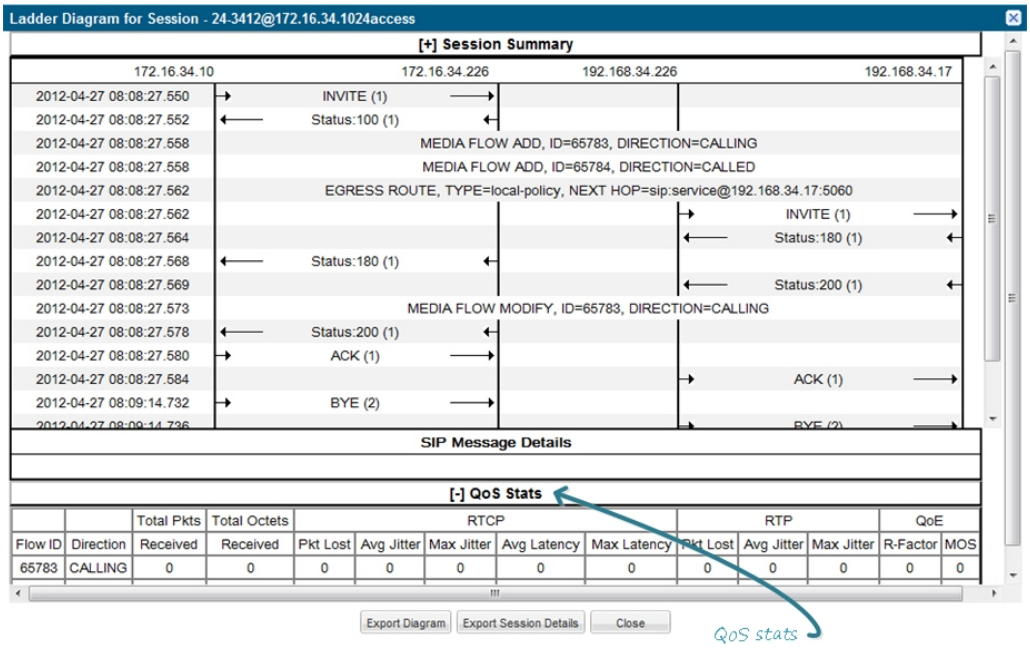
Display SIP Message Details

To display SIP Message Details:

1. On the Sessions Report page, click **Ladder diagram**, or select a record in the table and double-click on that record. The SIP Message Details window displays. This window displays the messages and status codes that occurred during the active call session or media event. You can use the information to troubleshoot calls and media events that failed or timed out when trying to connect.

QoS Statistics

The Quality of Service (QoS) window displays information about the quality of the service used on the call session or media event when the call or event was active.



Display QoS Statistics

To display QoS Statistics:

1. In the Ladder Diagram, click the **[+]** next to QoS Stats at the bottom of the Ladder Diagram window. The QoS window expands, and displays the QoS statistics for the call session or media event in focus. The following list describes each field in the QoS Statistics window.

Flow ID	ID number assigned to the call session or media event flow of data.
Direction	The direction of the call or media event flow. Valid values are: CALLING (egress direction) CALLED (ingress direction)
Total Pkts Received	Total number of data packets received on the interface during the active call session or media event.
Total Octets Received	Total number of octets received on the interface during the active call session or media event. An octet is a unit of digital information that consists of eight bits.
RTCP	Real-time Transport Control Protocol - used to send control packets to participants in a call.

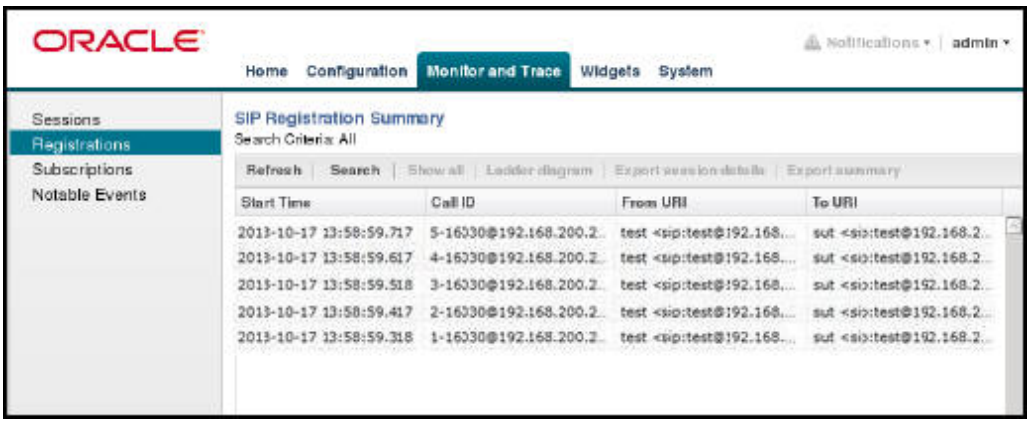
Pkts Lost	Number of RTCP data packets lost on the interface during the active call session or media event.
Avg Jitter	Average measure of the variability over time of the RTCP packet latency across a network. A network with constant latency has no variation (or jitter). Jitter is referred to as Packet Delay Variation (PDV). It is the difference in the one-way end-to-end delay values for packets of a flow. Jitter is measured in terms of a time deviation from the nominal packet interarrival times for successive packets.
Max Jitter	Maximum measure of the variability over time of the RTCP packet latency across a network. A network with constant latency has no variation (or jitter).
Avg Latency	Average observed one-way signaling latency during the active window period. This is the average amount of time the signaling travels in one direction.
Max Latency	Maximum observed one-way signaling latency during the sliding window period. This is the maximum amount of time the signaling travels in one direction.
RTP	Real-Time Transport Protocol - a standard packet format for delivering audio and video over the internet.
Pkts Lost	Number of RTP data packets lost on the interface during the active call session or media event.
Avg Jitter	Average measure of the variability over time of the RTP packet latency across a network. A network with constant latency has no variation (or jitter). Jitter is referred to as Packet Delay Variation (PDV). It is the difference in the one-way end-to-end delay values for packets of a flow. Jitter is measured in terms of a time deviation from the nominal packet interarrival times for successive packets.
Max Jitter	Maximum measure of the variability over time of the RTP packet latency across a network. A network with constant latency has no variation (or jitter).
QoE	Quality of Experience - measurement used to determine how well the network is satisfying the end user's requirements.
R-Factor	Average Quality of Service (QoS) factor observed during the active window period. Quality of service shapes traffic to provide different priority and level of performance to different data flows. R-factors are metrics in VoIP, that use a formula to take into account both user perceptions and the cumulative effect of equipment impairments to arrive at a numeric expression of voice quality. This statistic defines the call or transmission quality expressed as an R factor.
MOS	Mean Opinion Score (MOS) score. MOS is a measure of voice quality. MOS gives a numerical indication of the perceived quality of the media received after being transmitted and eventually compressed using Codecs.

2. Click **[-]** to close the QoS Stats window.

Registrations Report

The Registrations Report displays a summary of all logged SIP registrations sessions on the Oracle Enterprise Communications Broker (ECB.

The columns that display on the Registration Report page depend on the columns you selected in the "Customizing the Page Display" procedure.



The following table describes the columns on this page.

Heading	Description
Start Time	Timestamp of the first SIP message in the call session.
Call ID	Identification of the call source. Includes the phone number and source IP address.
To URI	URI formatted string that identifies the call destination information.
From URI	URI formatted string that identifies the call source information.
Local Expires	The current setting for the expiration of a registration request sent from the Integrated Media Gateway (IMG) to a Remote SIP User Agent. The default is 3600 sec.
Remote Expires	The current setting for the expiration of a registration request sent from the Remote SIP User Agent to the Integrated Media Gateway (IMG). The default is 3600 sec.
Ingress Realm	Incoming realm name.
Egress Realm	Outgoing realm name.

Heading	Description
Notable Event	Indicates if a notable event has occurred on the call session. Valid value is: local rejection - Sessions locally rejected at the E-SBC for any reason (for example, Session Agent (SA) unavailable, no route found, SIP signalling error, etc.); Session dialogue, capture media information and termination signalling; Any event flagged as a local rejection interesting event
Session ID	Identification assigned to the call session.
Ingress Src Addr	Source IP address of the incoming call or media event.
Egress Dest Addr	Destination IP address of the outgoing call or media event.
Request URI	Uniform Resource Identifier (URI) formatted string that identifies a resource via a protocol, name, location, and any other applicable characteristic, and is sent by the E-SBC in REQUEST headers.

The following table describes the buttons on this page.

Button	Description
	
Search	Allows you to specify parameters for performing a search for specific session summary records within the current report.
Show all	Displays all of the session summary records in the Session Report.
Ladder Diagram	Displays a Ladder Diagram of a specific record in the table. The Ladder Diagram displays detailed information about a call session or media event.
Export Session Details	Exports the SIP messages and media events associated with the selected session, to a file in text format on the local machine.
Export Summary	Exports all logged session summary records to a file in text format on the local machine.

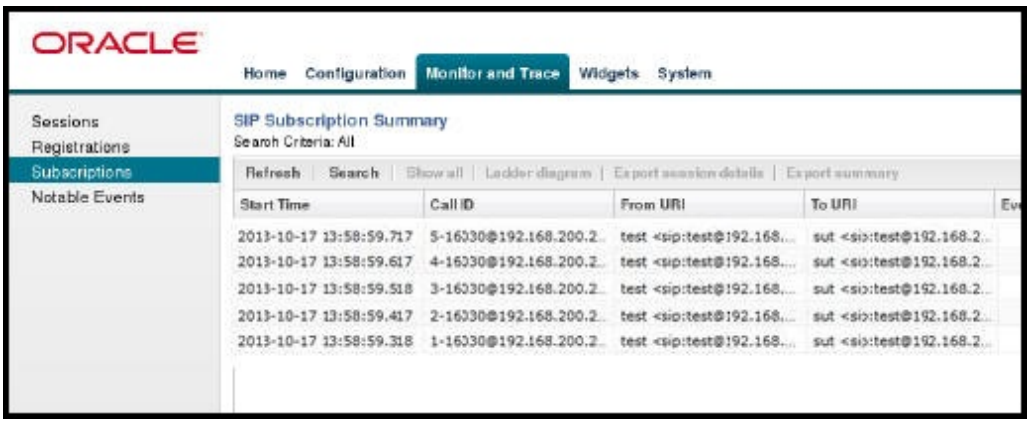
Display a Registrations Report

1. From the Web GUI, click **Monitor and Trace** > **Registrations**.
2. Use the buttons on the top of the page to view information about the records in this report.

Subscriptions Report

The Subscriptions Report displays a summary of all logged SIP subscription sessions on the Oracle Enterprise Communications Broker (ECB).

The columns that display on the Subscription Report page depend on the columns you selected in the procedure, "Customizing the Page Display."



The following table describes the columns on this page.

Heading	Description
Start Time	Timestamp of the first SIP message in the call session.
Call ID	Identification of the call source. Includes the phone number and source IP address.
From URI	URI formatted string that identifies the call source information.
To URI	URI formatted string that identifies the call destination information.

Heading	Description
Events	Specific subscribe event package that was sent from an endpoint to the destination endpoint. Applicable event packages can be: conference - Event package that allows users to subscribe to a conference Uniform Resource Identifier (URI). consent-pending additions - Event package used by SIP relays to inform user agents about the consent-related status of the entries to be added to a resource list. dialog - Event package that allows users to subscribe to another user, and receive notifications about the changes in the state of the INVITE-initiated dialogs in which the user is involved. message-summary - Event package that carries message-waiting status and message summaries from a messaging system to an interested User Agent (UA). presence - Event package that conveys the ability and willingness of a user to communicate across a set of devices. A presence protocol is a protocol for providing a presence service over the Internet or any IP network. reg - Event package that provides a way to monitor the status of *all* the registrations for a particular Address of Record (AoR). refer - Event package that provides a mechanism to allow the party sending the REFER to be notified of the outcome of a referenced request. winfo - Event package for watcher information. It tracks the state of subscriptions to a resource in another package. vq-rtcpx - Event package that collects and reports the metrics that measure quality for RTP sessions.
Local Expires	The current setting for the expiration of a registration request sent from the Integrated Media Gateway (IMG) to a Remote SIP User Agent. The default is 3600 sec.
Remote Expires	The current setting for the expiration of a registration request sent from the Remote SIP User Agent to the Integrated Media Gateway (IMG). The default is 3600 sec.
Ingress Realm	Incoming realm name.
Egress Realm	Outgoing realm name.
Notable Event	Indicates if a notable event has occurred on the call session. Valid value is: local rejection - Sessions locally rejected at the ECB for any reason (for example, Session Agent (SA) unavailable, no route found, SIP signalling error, etc.); Session dialogue, capture media information and termination signalling; Any event flagged as a local rejection interesting event
Session ID	Identification assigned to the call session.

Heading	Description
Ingress Src Addr	Source IP address of the incoming call or media event.
Egress Dest Addr	Destination IP address of the outgoing call or media event.
Request URI	Uniform Resource Identifier (URI) formatted string that identifies a resource via a protocol, name, location, and any other applicable characteristic, and is sent by the ECB in REQUEST headers.

The following table describes the buttons on this page.

Button	Description
	
Search	Allows you to specify parameters for performing a search for specific session summary records within the current report.
Show all	Displays all of the session summary records in the Session Report.
Ladder Diagram	Displays a Ladder Diagram of a specific record in the table. The Ladder Diagram displays detailed information about a call session or media event.
Export Session Details	Exports the SIP messages and media events associated with the selected session, to a file in text format on the local machine.
Export Summary	Exports all logged session summary records to a file in text format on the local machine.

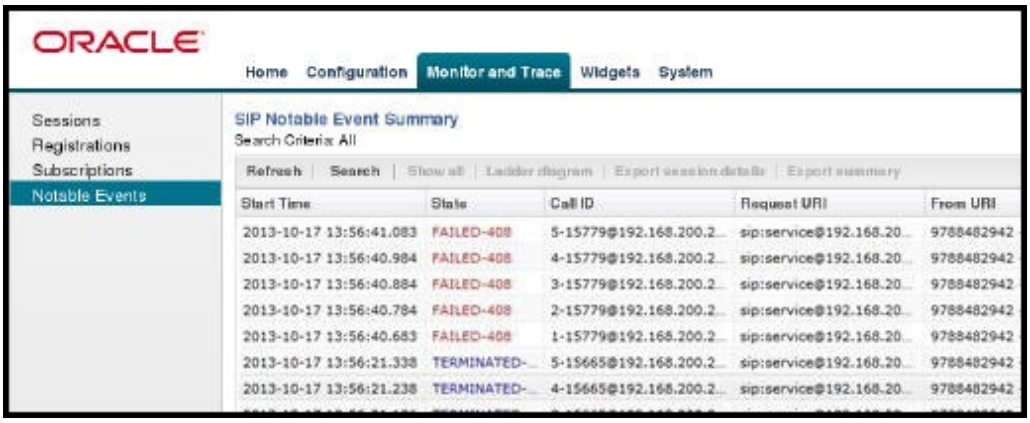
Display a Subscriptions Report

1. From the Web GUI, click **Monitor and Trace** > **Subscriptions**.
2. Use the buttons on the top of the page to view information about the records in this report.

Notable Events Report

The Notable Events Report contains all logged sessions with a notable event associated with the session on the Oracle Enterprise Communications Broker (ECB).

The columns that display on the Notable Events Report page depend on the columns that you selected in the procedure, "Customizing the Page Display."



The following table describes the columns that this page can display.

Heading	Description
Start Time	Timestamp of the first SIP message in the call session.
State	Status of the call or media event session. Valid values are: INITIAL Session for which an INVITE or SUBSCRIBE was forwarded. EARLY Session received the first provisional response (1xx other than 100). ESTABLISHED Session for which a success (2xx) response was received. TERMINATED Session that has ended by receiving or sending a BYE for an “Established” session or forwarding an error response for an “Initial” or Early session. The session remains in the terminated state until all the resources for the session are freed up. FAILED Session that has failed due to a 4xx or 5xx error code.
Call ID	Identification of the call source. Includes the phone number and source IP address.
Request URI	Uniform Resource Identifier (URI) formatted string that identifies a resource via a protocol, name, location, and any other applicable characteristic, and is sent by the ECB in REQUEST headers.
From URI	URI formatted string that identifies the call source information.
To URI	URI formatted string that identifies the call destination information.

Heading	Description
Notable Event	Indicates if a notable event has occurred on the call session. Valid values are: short session - Sessions that don't meet a minimum configurable duration threshold; Session dialogue, captured media information and termination signalling; Any event flagged as a short session interesting event. local rejection - Sessions locally rejected at the ECB for any reason (for example, Session Agent (SA) unavailable, no route found, SIP signalling error, etc.); Session dialogue, capture media information and termination signalling; Any event flagged as a local rejection interesting event.
Session ID	Identification assigned to the call session.
Ingress Src Addr	Source IP address of the incoming call or media event.
Ingress Src Port	Source port of the incoming call or media event.
Egress Dest Addr	Destination IP address of the outgoing call or media event.
Egress Dest Port	Destination port of the outgoing call or media event.
Object ID	ID number of the object in a row. Use to aid troubleshooting.

The following table describes the buttons on this page.

Button	Description
	
Search	Allows you to specify parameters for performing a search for specific session summary records within the current report.
Show all	Displays all of the session summary records in the Session Report.
Ladder Diagram	Displays a Ladder Diagram of a specific record in the table. The Ladder Diagram displays detailed information about a call session or media event.
Export Session Details	Exports the SIP messages and media events associated with the selected session, to a file in text format on the local machine.
Export Summary	Exports all logged session summary records to a file in text format on the local machine.

Display a Notable Events Report

1. From the Web GUI, click **Monitor and Trace** > **Notable Events**.

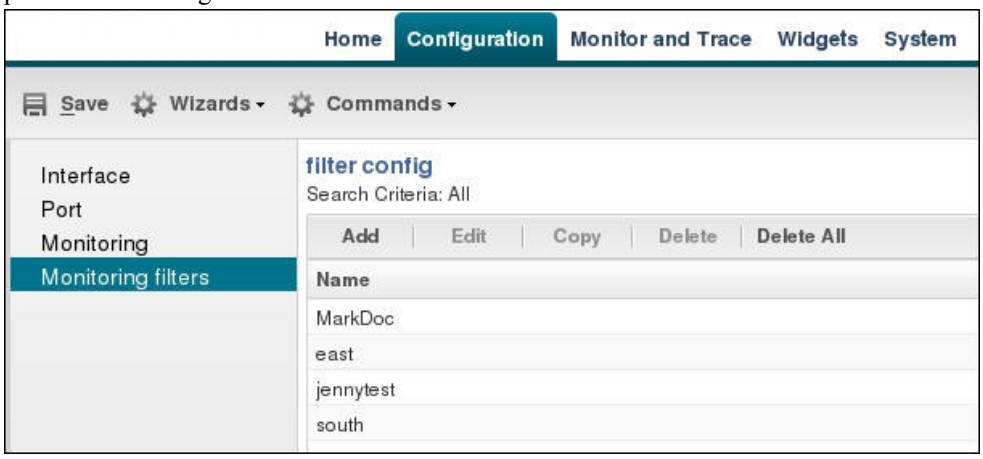
2. Use the buttons on the top of the page to view information about the records in this report.

SIP Monitor and Trace Filter Configuration

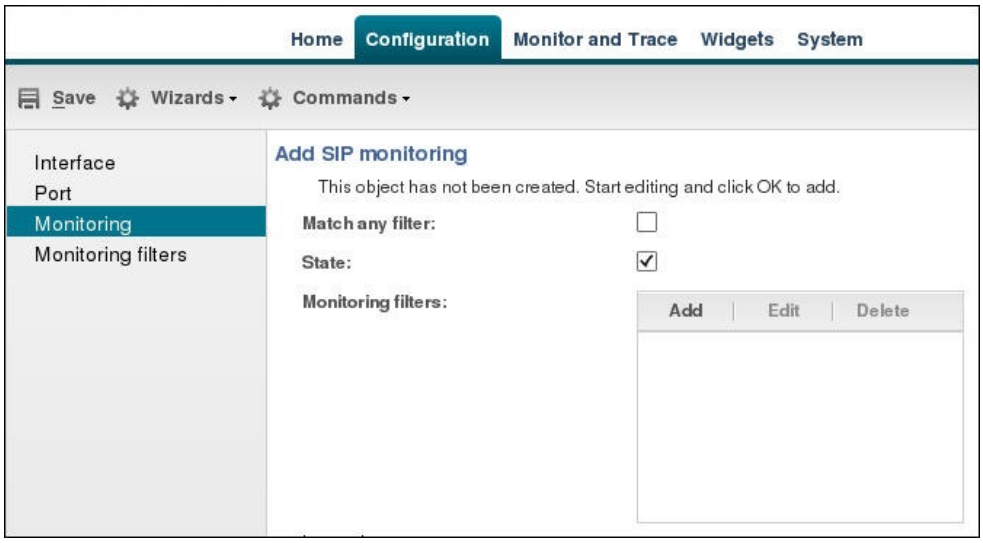
The SIP Monitor and Trace function allows you to monitor SIP sessions for notable events and display the results in the Oracle Enterprise Communications Broker (ECB) SIP Notable Events summary. Such information may help you perform troubleshooting. For more targeted monitoring, you can configure filters on particular users and addresses on the ECB, and on a specific agent.

As of PCZ200M4, the ECB includes the following changes:

- The SIP Interface configuration page adds the **Monitoring filters** object to the navigation pane. Use to configure individual filters.



- The pre-existing **Monitoring** object on the SIP interface configuration page adds the **Monitoring filters** element to the dialog. Use to apply filters to the ECB.



- The Add Agents configuration page adds the**Monitoring filters** configuration element to the Advanced section. Use to apply filters to an agent.

- When you upgrade to PCZ200M4, note that the system does not support the former "Enable SIP Monitor and Trace" setting. You must re-configure SNMP event traps through the dialogs described above. See "Caveats" for more information.

Use the following filter configuration process for both new installations and upgrades.

1. Create one or more filters in the Monitoring Filters object. You may use an asterisk character as a filter, if you want to monitor all session data.
2. Add one or more filters to the Monitoring object.
3. (Optional) Add one or more monitoring filters to an agent that you want to monitor.

Search for a Record

The **Search** button at the top of the report page allows you to find a specific record within a Monitor and Trace report. It also allows you to specify criteria on which to perform the search.

After defining a search criteria in the Search Filter dialog box, clicking Search automatically populates the report page with the records that match the specified criteria specified. The search performs the filtering process of criteria dependent on the report page from which you are running the search.

For example, performing a search from the Sessions report page displays only the reports pertaining to call sessions. If you perform a search on the Registration report page, only the reports pertaining to call registrations displays on the report page. The search string containing the criteria on which you performed the search, displays in the top left corner of the page.

> **Note:**
>
> A SIP Monitor and Trace global search can find items in the SIP headers, as well. The system saves the search criteria until you click **Reset** in the dialog box, or until you log out of the HTTP session.

Perform a Search

To perform a search:

You can specify a value for any or all of the fields in the Search box. The search process searches for records with all of the values you specify and displays only the records with these

values. If you perform a “Global Search”, AND specify values in other fields, the search process searches the other specified fields first and then filters on the “Global Search” field.

If you specify a “*” in a search string, the search is performed on that exact string. For example, if you search for “123*45”, the search shows results for all strings containing “123*45”.

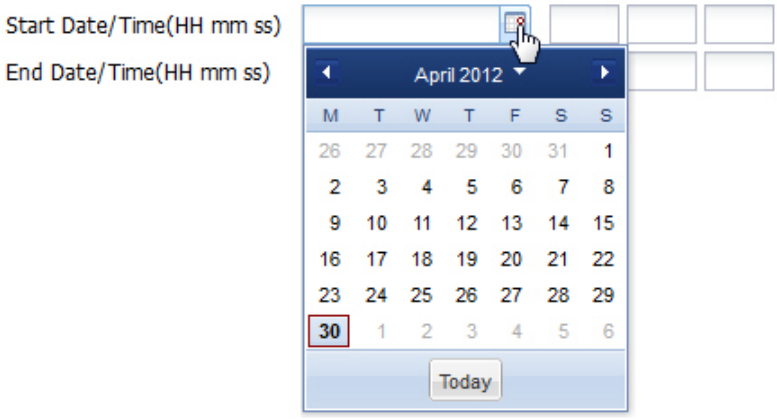
You can use quotes (“ “) to specify a search. For example, you can enter Smith and the search finds all of the records that match Smith, such as: John **Smith**field<sip:sipp@192.168.1.70:5070>;tag=12260SIPpTag001.

If you enter a space before or after a quotation mark, (for example, “Smith “), the search returns no data.

1. In any reports page, click **Search**.
2. In the Global Search field, specify a string to search all parameters in all records. Valid values are alpha-numeric characters.

> **Note:**
>
> The Global Search option searches all parameters in all the session records stored in memory. All values you specify in other fields are searched before the value specified in the Global Search field is used.

3. In the From URI field, enter the URI formatted string of the call source information you are searching. Valid values are alpha-numeric characters. For example, sipp<sip:sipp@172.16.34.10:5060;tag=24.
4. In the Requested URI field, enter the URI formatted string that contains a protocol, name, location, or any other applicable characteristic, that is sent by the Net-Net ECB in the REQUEST header. Valid values are alpha-numeric characters. For example, sip:service@172.16.34.226:5060.
5. In the To URI field, enter URI formatted string of the call destination information you are searching. Valid values are alpha-numeric characters. For example, sut<sip:service@172.16.34.226:5060;tag=99.
6. In the Start Date/Time (HH mm ss) field, enter a starting date to search on in the first text box in the format YYYY-MM-DD (where Y =year, M=month, and D=day). or Click on the calendar icon in this field to display a calendar from which you can select a date. Navigate the calendar to find the date you want and click on it to enter it into this field, or click <**Today**> to enter today’s date. For example, 2012-04-15 would search for all records starting on April 15, 2012. Valid values are numeric characters only. Enter a start time to search on in the last three text boxes in the format HH mm ss (where H=hour, m=minutes, and s=seconds). For example, 01 30 45 would search for all records starting at 1:30 and 45 seconds. Valid value are numeric characters only.



7. In the End Date/Time (HH mm ss) field, repeat the process of entering a date and time as provided in Step 7.
8. To search on additional parameters, click on the Additional Identifiers down arrow to expand the dialog box.

Specify Additional Identifiers

To specify additional identifiers:

1. In the Session Id field, enter the ID of the call session you want to search. Valid values are alpha-numeric characters. For example, 22-3412@172.16.34.1.
2. In the In Call ID field, enter the ID of the incoming call (phone number and source IP address). Valid values are alpha-numeric characters. For example, 25-3412@172.16.34.10.
3. In the Out Call ID field, enter the ID of the outgoing call (phone number and IP address). Valid values are alpha-numeric characters. For example, 14-3412@172.14.54.6.
4. In the State (with result code) field, enter the status of the call session with the result code for which you want to search. Valid values are (case-sensitive):
   - INITIAL-<result code>
   - EARLY-<result code>
   - ESTABLISHED-<result code>
   - TERMINATED-<result code>
   - FAILED-<result code>

   Result codes can range from 1xx to 5xx. For example, terminated-200, or failed-400.
5. In the Notable Event field, select the notable event for which you want to search. Valid values are:
   - any-event - search displays any notable event that was stored in memory.
   - short-session - search displays only records that indicate a short-session duration has occurred.
   - local-rejection - search displays only records that indicate a local-rejection has occurred.
6. To search on additional parameters, click on the Additional Search Options down arrow to expand the dialog box.

Specify Additional Search Options

To specify additional search options:

1. In the "**In Realm**" field, enter the name of the realm for which the incoming call belongs. Valid values are alpha-numeric characters. For example, access.
2. In the "**Out Realm**" field, enter the name of the realm for which the outgoing call belongs. Valid values are alpha-numeric characters. For example, backbone.
3. In the "**In SA**" field, enter the name of the session agent (SA) on the incoming call session. Valid values are alpha-numeric characters. For example, SA1.
4. In the "**Out SA**" field, enter the name of the session agent (SA) on the outgoing call session. Valid values are alpha-numeric characters. For example, SA2.
5. In the "**In Source Addr**" field, enter the source IP address of the SA that accepted the incoming call session. IP Address must be entered in dotted decimal format (0.0.0.0). For example, 172.45.6.7.
6. In the "**Out Dest Addr**" field, enter the destination IP address of the SA that accepted the outgoing call session. IP Address must be entered in dotted decimal format (0.0.0.0). For example, 172.64.56.7.
7. In the In Network Interface field, enter the incoming core network interface that connects the Net-Net ECB to your network. IP Address must be entered in dotted decimal format (0.0.0.0). For example, 192.45.6.7.
8. In the Out Network Interface field, enter the outgoing network interface that connects your Net-Net ECB to the outside network. IP Address must be entered in dotted decimal format (0.0.0.0). For example, 192.45.6.8.
9. Click <**Search**> to perform the search with the values you specified. A list of the records that the search process filtered, display in the window. The GUI saves the search specifications until you click <**Reset**> in the search dialog box, OR until you log out of the GUI.

Exporting Information to a Text File

Monitor and Trace allows you to export information to a text file from the Sessions, Registrations, Subscriptions and Notable Events Reports, as well as from a specific ladder diagram, or from a page containing the results of a search. The system exports data to a file that you can open and view as required.

You can export any of the following:

- All information from each report
- Information from a specific record only
- Information from a search result
- Information from a Ladder Diagram

The following table identifies the buttons to use to export specific information from Monitor and Trace. All the export buttons in the GUI export to text files.

Button	Description
From the Sessions, Registrations, Subscriptions, and Notable Events Reports:	

Button	Description
Export Session Details	Exports the SIP messages and media events associated with the selected session, to a file in text format on the local machine.
Export Summary	Exports all logged session summary records to a file in text format on the local machine. Note: This button exports ALL call session summary records or the records that matched a search criteria to the file.
From the Ladder Diagram:	
Export Diagram	Exports all of the information in the Ladder Diagram (Session Summary, SIP Message Details, and QoS statistics), to an HTML file format on the local machine.
Export Session Details	Exports detailed information about the SIP messages and media events in the Ladder Diagram associated with the selected session, to a file in text format on the local machine.

Export Report Information to a Text File

To export information from a Monitor and Trac report to a text file:

> **Note:**
>
> The GUI exports Ladder Diagrams as HTML files.

1. From the Web GUI, click the **Monitor and Trace** tab.
2. On the Monitor and Trace page, select a report type. For example, Subscriptions.
3. On the report Summary page, select a report from the list, and do one of the following:
   - Click **Export session details**.
   - Click **Export summary**.
4. In the SessionDetails.txt or SummaryExport.txt dialog, do one of the following:
   - Click **Open with**, and select the application with which to open the resulting text file.
   - Click **Save file** to save the text file to your local PC.
5. Click **OK** to export the report information.

15
Troubleshooting and Maintenance

Whereas the Oracle Enterprise Communications Broker Broker Administrator's Essentials Guide provides system-based troubleshooting information, this chapter presents parallel information related to the SIP services the Oracle Enterprise Communications Broker provides.

To a large extent, the Oracle Enterprise Communications Broker's widgets display status and quantitative information about SIP traffic. Many of these widgets provide information that is self-explanatory. This chapter provides descriptions and instructions on key widgets used to analyze service operations.

Maintenance consists of a variety of tasks, including managing system files. The Oracle Enterprise Communications Broker's System tab provides access to file management controls, and is described in this chapter.

Refer to the Oracle Enterprise Communications Broker Broker Administrator's Essentials Guide for further troubleshooting and maintenance information related to system administration.

Audit Logs

The Oracle Enterprise Communications Broker (ECB) can record user actions in audit logs by way of the Web GUI. The audit logs record the creation, modification, and deletion of all user-accessible configuration elements, as well as attempted access to critical security data such as public keys. For each logged event, the system provides the associated user-id, date, time, event type, and success or failure data.

You can configure the system to record audit log information in either verbose mode or brief mode. Verbose mode captures the system configuration after every change, and displays both the previous settings and the new settings in addition to the event details. Brief mode displays only the event details. Although you can specify the recording mode, you cannot specify which actions the system records. The following table lists the actions that the system records.

Source	Actions Recorded
Global	• Log on and log off. • Save a template configuration. • Click **Complete** in a Wizard.
Home tab	• Add, reset, and save. • Change Widget settings.
Configuration tab	• Save and activate a configuration. • Discard a configuration. • Add, edit, delete, and copy configuration changes. • Run the generate and import certificate commands.

Source	Actions Recorded
Widgets tab	• Export from a Widget. • Add a Widget to favorites. • Clear, clear all on alarm, add, and delete license.
System tab	• Add audit entries to the system file management actions, such as upload, download, restore, backup, add, edit, and delete. • Force an HA switch over. • Run the Show Support Information command. • Run the Upgrade Software wizard. • Download and view an audit log.
Monitor and Trace tab	• Export the summary. • Export the session detail.

The system writes audit log events in Comma Separated Values (CSV) lists in the following format:

```
{TimeStamp,
src-user@address:port,Category,EventType,Result,Resource,Prev,
Detail}
```

The following table describes each value written to an audit log event.

Log Element	Information Provided
TimeStamp	Shows the time when the system wrote the event to the audit log.
src-user@address:port	Identifies the system that wrote the audit log line.
Category	Classifies the event as: • Configuration • Security • System
EventType	Identifies the action that caused the event as: • Activate-config • Acquire-config • Create • Data-access • Delete • Halt • Login • Logout • Modify • Reboot • Save-config
Result	Identifies the outcome of the event as: • Failure • Success

Log Element	Information Provided
Resource	Describes the action within the event. Some of the numerous actions that the system can log include: • Authentication • Banner (Means that someone edited the log on banner text.) • Download <filename> • Generate public key • Reboot • Upload <filename>
Prev—(verbose mode)	Displays the setting prior to this change.
Details—(verbose mode)	Displays additional information about the change, depending on the following event types: • Create—displays “New = element added." • Data-access—displays “Element = accessed element.” • Delete—displays “Element = deleted element.” • Modify—displays “Previous = oldValue New = newValue.”

As the ECB records audit log data, users with admin privileges can read, copy, and download that information from the Web GUI. No one can delete or edit the original log. You can View, Refresh, and Download audit logs by way of the System tab. When you click File Management, the system displays the File Type drop-down list, which includes "Audit Log" as a selection.

You can configure the system to transfer audit log files to an SFTP server by way of secure FTP push, when conditions satisfy one of the following specifications.

- The specified amount of time since the last transfer elapsed.
- The size of the audit log reached the specified threshold. (Measured in Megabytes)
- The size of the audit log reached the specified percentage of the allocated storage space.

The ECB transfers the audit logs to a designated directory on the target SFTP server. The audit log file is stored on the target SFTP server with a filename in the following format: **audit<timestamp>**. The timestamp is a 12-digit string the YYYYMMDDHHMM format.

Use the following process to configure transferring audit logs to an SFTP server.

1. Configure secure FTP push. See "Secure FTP Push Configuration."
2. Configure audit logging. See "Configure Audit Logging."

Secure FTP Push Configuration

You can configure the Oracle Enterprise Communications Broker (ECB) to securely send audit log files to an SFTP push receiver for storage. Configure secure FTP push before you configure audit logging.

You can configure the Oracle Enterprise Communications Broker (ECB) to log on to a push receiver using one of the following authentication methods to create a secure connection.

Password
Configure a username and password, and leave the **public-key** parameter blank. Note that you must also import the host key from the SFTP server to the ECB for this type of authentication.

Public key
Set the **public-key** parameter to a configured public key record name including an account **username**, and configure the SFTP server with the public key pair from the ECB.

It is also common for the SFTP server to run the Linux operating system. For Linux, the command ssh-keygen-e creates the public key that you need to import to the ECB. The ssh-keygen-e command sequence requires you to specify the file export type, as follows.

```
[linux-vpn-1 ~]# ssh-keygen -e
Enter file in which the key is (/root/.ssh/id_rsa/): /etc/ssh/
ssh_host_rsa_key.pub
```

If you cannot access the SFTP server directly, but you can access it from another Linux host, use the ssh-keyscan command to get the key. An example command line follows.

```
root@server:~$ssh-keyscan -t dsa sftp.server.com
```

Configure Secure FTP Push with Public Key Authentication

For increased security when sending files from the Oracle Enterprise Communications Broker (ECB) to an SFTP server, you can choose authentication by public key exchange rather than by password. To use a public key exchange, you must configure public key profiles on both devices and import the key from each device into the other.

The following list of tasks shows the process for configuring authentication by public key between the ECB and an SFTP server. For each step in the process, see the corresponding topic for detailed instructions.

1. Generate an RSA public key on the ECB. See "Generate an RSA Public Key."
2. Create a DSA public key on the SFTP server. See "Generate a DSA Public Key."
3. Import the DSA public key from the SFTP server into the ECB using the **known-host** option in the Import Key dialog. See "Import a DSA Public Key."
4. Add the RSA public key to the authorized_keys file in the .ssh directory on the SFTP server. See "Copy the RSA Public Key to the SFTP Server."

Generate an RSA Public Key

Add a public key profile on the Oracle Enterprise Communications Broker (ECB) and generate an RSA key. You will later import the RSA key into the SFTP server to enable authentication by way of public key exchange with the ECB.

1. From the Web GUI, click **Configuration** > **Security** > **Public key**.
2. On the Public Key page, click **Add**.
3. In the Add Public Key dialog, do the following:

Attributes	Instructions
Name	Enter the name of this profile.
Type	Select RSA.

Attributes	Instructions
Size	Enter one of the following: • 512 • 1024 • 2048 • 4096

4. Click **OK** to create the public key profile.

   The system displays the Public Key list box including the new profile.

5. Save and activate the configuration.
6. Select the newly created profile, and click **Generate key**.

   The ECB displays the key in the Generate Key text box for you to copy to the SFTP server.

7. Save the configuration.

- Generate a DSA public key.

Generate a DSA Public Key

Generate and save a DSA public key on the SFTP server. You will later import the DSA key into the Oracle Enterprise Communications Broker (ECB) to enable authentication by way of public key exchange with the SFTP server.

1. Run the following command on the SFTP server:
   ssh-keygen -e -f /etc/ssh/ssh_host_dsa_key.pub | tee sftp_host_dsa_key.pub
2. Save the key to the authorized_keys file in the .ssh directory on the SFTP server.

- Import the DSA key into the ECB.

Import a DSA Public Key

Import a DSA public key from the SFTP server into the Oracle Enterprise Communications Broker (ECB).

- Generate and save a DSA public key on the SFTP server.

Perform the following procedure on the ECB and select "known-host" for type.

1. Access the SSH file system on the SFTP server by way of a terminal emulation program.
2. On the SFTP server, copy the base64 encoded public file. Be sure to include the Begin and End markers, as specified by RFC 4716 *The Secure Shell (SSH) Public Key File Format*.

   For OpenSSH implementations host files are generally found at `/etc/ssh/ssh_host_dsa_key.pub`, or `/etc/ssh/sss_host_rsa.pub`. Other SSH implementations can differ.

3. On the ECB, click **Configuration** > **Security** > **Public Key**.
4. On the Public key page, click **Import key**, and do the following.

Attributes	Instructions
Type	Select known-host.
Name	Enter a name for your profile, which the ECB displays in public key drop-down lists.

Attributes	Instructions
SSH public key	Paste the DSA public key from the SFTP server into the text box. Ensure that the text of the key ends with a semi-colon.

5. Click **Import**.

   The ECB imports the key and makes it available for configuration as the public key on an external device.

Copy the RSA public key to the SFTP server.

Copy the RSA Public Key to the SFTP Server

Copy the RSA public key from the from the Oracle Enterprise Communications Broker (ECB) to the authorized_keys file in the .ssh directory on the SFTP server.

- Confirm that the .ssh directory exists on the SFTP server.
- Confirm the following permissions: Chmod 700 for .ssh and Chmod 600 for authorized_keys.

When adding the RSA key to the authorized_keys file, ensure that no spaces occur inside the key. Insert one space between the ssh-rsa prefix and the key. Insert one space between the key and the suffix. For example, ssh-rsa <key> root@1.1.1.1.

1. Access the SSH file system on a configured SFTP server with a terminal emulation program.
2. Copy the RSA key to the SFTP server, using a text editor such as vi or emacs, and paste the RSA key to the end of the authorized_keys file.

Configure Audit Logging

The Oracle Enterprise Communications Broker (ECB) provides a means of tracking user actions through Audit Logs. You can specify how the system records audit log information, and where to send the logs for archiving. You can configure the system to record in either brief or verbose mode. Verbose mode captures the system configuration after every change, and displays both the previous and new settings in addition to the event details. Brief mode displays only the event details.

- Configure one or more push receivers to receive the audit logs. See the documentation for the receiver.
- If you want to use public keys for authentication between the ECB and the push receiver, configure public key profiles on both devices before configuring audit logging. See "Configure Secure File Transfer with Public Keys."

1. Log on to the ECB, and click **Configuration** > **Security** > **Admin-Security** > **Audit Logging**.
2. On the Audit Logging page, do the following:

Attributes	Instructions
State	Select to enable event recording in the audit log.
Detail level	Select brief (default) or verbose output.

Attributes	Instructions
File transfer time	Specify the amount of time, in hours, from the completion of the last transfer to the beginning of the next transfer. This determines when a file transfer occurs unless the Max storage space or Max file size triggers the transfer first. • Minimum: 0, which disables this file transfer time function. • Maximum: 65535 • Default: 720
Max storage space	Specify the maximum amount of space that the audit log can consume on the ECB in MB. • Minimum: 0 • Maximum: 32 (default)
Percentage full	Use in conjunction with Max storage space to specify the percent of the Max storage space that triggers file transfer. This determines when a file transfer occurs unless the File transfer time or Max file size triggers the transfer first. • Minimum: 0, which disables this percentage full function. • Maximum: 99 • Default: 75
Max file size	Set the maximum size in Mega Bytes that the audit log can be before the system transfers the file. This determines when a file transfer occurs unless the Max storage space or Max file size triggers the transfer first. • Minimum: 0, which disables this maximum file size function. • Maximum: 10 • Default: 5

Attributes	Instructions
Push receiver	Add a push receiver and configure the following parameters for sending audit log files from the ECB to the receiver: • Server—Enter the IP address of the FTP/SFTP server to which you want the ECB to push audit log files. Default: 0.0.0.0. • Port—Enter the port number on the FTP/SFTP server to which the ECB will send audit log files. Range:1-65535. Default: 22 • Remote path—Enter the pathname to send the audit log files to the push receiver. Files are placed in this location on the FTP/SFTP server. Value: <string> remote pathname. • Filename prefix—Enter the filename prefix to prepend to the audit log files that the ECB sends to the push receiver. The ECB does not rename local files. Values: <string> prefix for filenames. • Username—Enter the username the ECB uses to connect to this push receiver. • Auth type—Select the authentication methodology. Password (default) or public key. • Do one of the following: Password—If you set the Auth type to password, click **Set** to enter and confirm the password used to access this push receiver. Public key—If you set the Auth type to public key, select the public key profile that you want from the drop-down list.

3. Click **OK**.
4. Save the configuration.

Key Widgets

The section presents explanations and instructions on the use of key widgets available with the Oracle Enterprise Communications Broker that provide status and other operational information about service data. These widgets are available from the **Widgets** tab, as well as the system Dashboard. Generic access and usage information is presented in the Getting Started chapter.

Agent Status Widget

There is a widget for displaying the status of session-agents added to the web GUI. The widget is available from the Statistics Portal.

The columns of each row displays both configured and runtime information.

- Name
- IP Address
- State
- Active Calls - inbound

- Active Calls - outbound
- Round-Trip-Time

Because the number of session-agents could be very large, the use of paging, searching, and sorting is utilized.

User Management Interface

Grid widget

Name	IP Address	State	Active Calls (Inbound)	Active Calls (Outbound)	RTT (ms)
SBC 1	192.168.1.1	IN-SERVICE	1000	1000	10
SBC 2	192.168.1.2	OUT-OF-SERVICE	0	0	0
SBC 3	172.16.1.1	IN-SERVICE	30	30	123

Performance

This Widget will not impact signaling performance. However it could become unresponsive.

Broker Lookup Widget

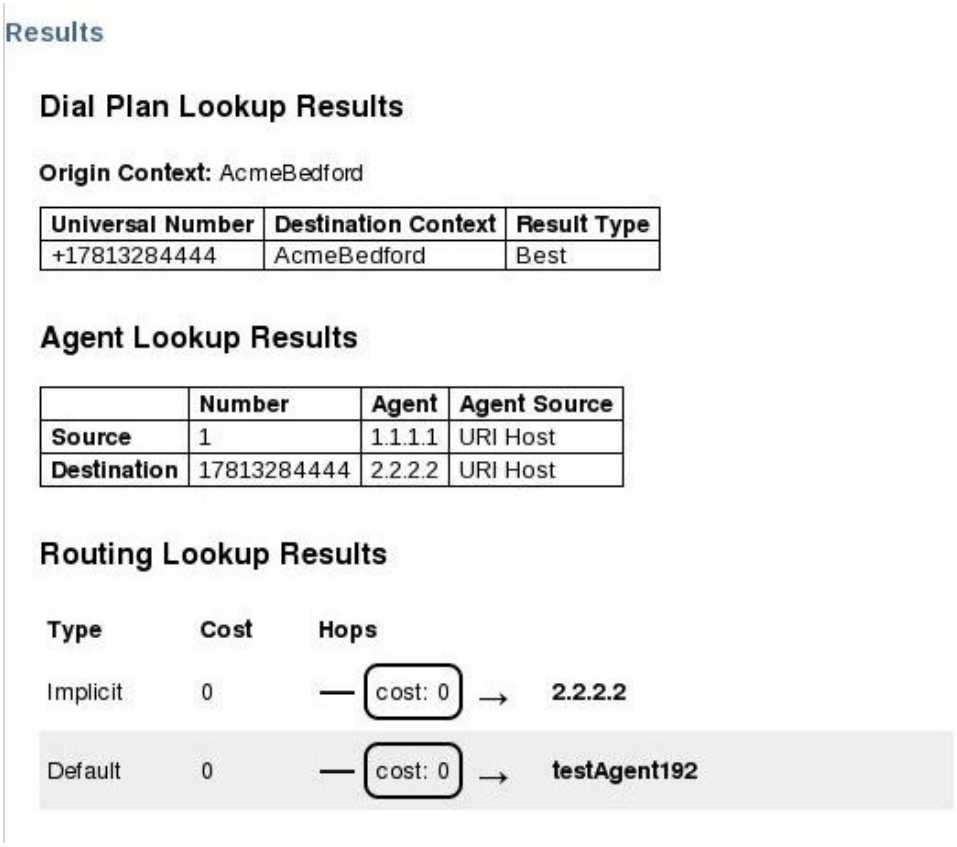
There is a widget for performing a broker lookup added to the web GUI. The widget is used to test how a given call is processed, based on the current dial plan, agent and routing configuration. The user enters the call's FROM_URI, REQUEST_URI and Source Agent and clicks the **Begin** button to execute the test. The widget displays all possible ways that a call with those parameters could be processed, including all hops and their associated costs.

Requirements

This command widget uses the Settings Panel and the Results Panel. In the Settings Panel, the user enters the From-URI and Request-URI information. Source Agent information can also be selected. The results of the lookup are displayed in the results window.

Result Analysis

The diagram below displays the results fields of the Broker Lookup widget interface.



Result descriptions are provided in the lists below, based on category and column.

Dial Plan Lookup Results

- Universal Number—This is the complete universal (usually E.164) called number that the Dial Plan engine found for the information entered by the user.
- Destination Context—This is the Dialing Context associated with the universal number.
- Result Type—This indicates the certainty of the result. There are 3 possible values:
  - Best—The result is either the only result found, or is the best match. This is the universal number that will be called.
  - Ambiguous—This dial plan result conflicts with other dial plan results. The Oracle Enterprise Communications Broker is not able to route the call because the intended destination is unclear. If a result is ambiguous, there are other ambiguous results as well, and no result can be labeled as 'Best'.
  - None—This dial plan result does not conflict with others, but it is also not the best result, so it will not be used. Another result is listed as 'Best'.

Agent Lookup Results

- Unlabeled column—Indicates whether the row refers to the source or destination agent.
- Number—This is the calling or called number, respectively, that was used to look up the agent.
- Agent—This is the agent that was found for this number.
- Agent Source—This is the source from which the agent was found. The Oracle Enterprise Communications Broker tries to find the agent associated with this number using the following methods in order:

  - User DB
  - LDAP
  - URI Host (taken from the From-URI)
  - Inbound Session-Agent (only applies to source agent)

Routing Lookup Results

- Each route can be one of the following types:
  - Implicit—Routing directly to the given Request-URI.
  - Configured—A route that was configured by the user.
  - Default—A route that was configured by the user that matches any incoming call. For example, a route that has all fields configured as asterisks.
- This is the cost based on the routing configuration.
- This is a visual diagram of the route, in which each individual hop is displayed as bold text, and the cost incurred by routing the call to each given hop is displayed as a box with an arrow through it.

Performance

This Widget does not impact signaling performance. However it could become unresponsive.

Connectivity Tester Widget

There is a widget for testing connectivity to various agents added to the web GUI. The widget allows the user to perform various types of Pings.

Supported Pings.

- SIP OPTIONS ping
- SIP INFO ping

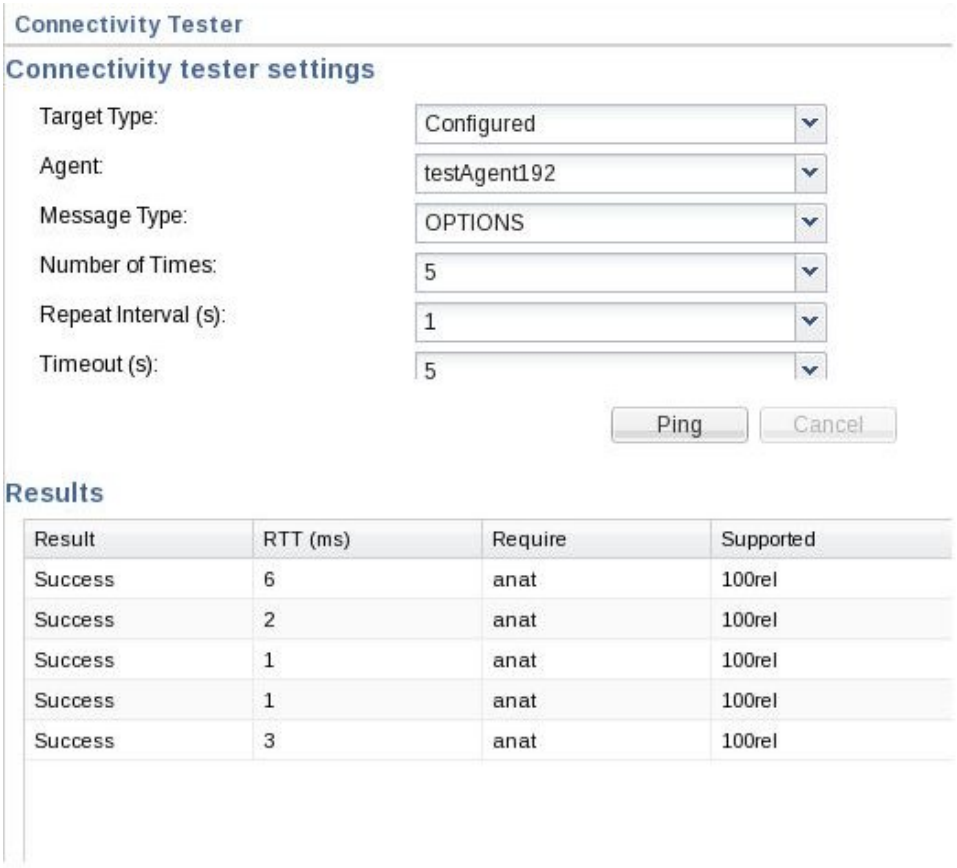
This widget is available from the Widgets Tab.

Requirements

The widget uses a Settings Panel and a Results Panel. In the Settings Panel the user can configure various options for the Ping command. The reply is displayed in the Results Panel.

User Management Interfaces

Widget layout



Settings and result descriptions are provided in the lists below, based on category and column.

Connectivity tester settings

- Target Type—This field specifies the type of target to which the the system sends the test. There are two options:
  - Configured—Setting target type to 'configured' causes the system to populate the subsequent field (agent) with a drop-down list of configured agents.
  - URI—Setting target type to 'URI' formats the agent field to text, allowing the user to enter any valid URI in the agent field.
- Agent—This field is dependent on the Target type field described above.
- Message Type—This field specifies the type of SIP message to be used for the test. Options include:
  - OPTIONS
  - INFO
- Number of Times—This field specifies
- Repeat Interval(s)—This field specifies the length of time between repeats.
- Timeout(s)—This field specifies the length of time to wait for each ping to respond before timing out.

Results

- Result—This column displays the result of the test, either success or failure.

- RTT(ms)—This column displays the tound trip time of the test, in milliseconds.
- Require—This column displays the protocols required by the pinged agent as supplemental test results. These protocols are listed in the Require header of the response.
- Supported—This column displays the protocols supported by the pinged agent as supplemental test results. These protocols are listed in the Supported header of the response.

Performance

This widget does not impact Signaling performance. However it could become unresponsive.

Registration Cache Dashboard Widget

There is a dashboard widget for displaying registration cache entries. The registration cache is implemented in the SIP tasks. The dashboard widgets back-end is implemented in web tasks. The SIP task will walk all SIP users, using the new users SPL API. The entire registration cache is displayed. This information will be packaged and sent back to the web task. The response will be parsed in the web task and returned to the user in the form of a widget. Multiple (AOR) contacts are displayed with the same user name, different contacts.

Widget Columns

SIP registration cache table.

User	Address of record
Universal number	ECB E.164 number mapping (+17811234567)
Contact endpoint	SIP contact
Contact expire endpoint	Expires (end point side)
Contact expire	Expires (real)

Performance

This widget will not impact signaling performance. However it could become unresponsive.

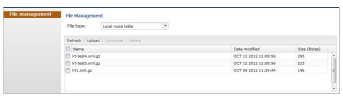
System File Management

Basic description of GUI file management dialogs.

The System tab within the GUI provides an easy and convenient means for managing your Oracle Enterprise Communications Broker system files. It allows you to perform the following:

- Upload files
- Download files
- Delete files
- Restore files

The following is an example of the System tab window.



The following table identifies the files you can manage on the Oracle Enterprise Communications Broker.

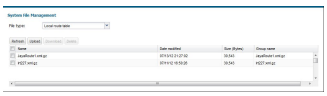
File Type	Format	Description
Local subscriber table (LST)	.xml	Local subscriber table (LST) file that you can apply to the Net-Net ECB. The LST is an in-memory table that contains subscriber information needed to register users.
SPL Plug-in	.lua	Session Plug-in Language (SPL) file that you can apply to the Oracle Enterprise Communications Broker to incorporate additional functionality. The SPL file contains a programming language that is capable of performing various tasks by utilizing APIs and callbacks in the Oracle Enterprise Communications Broker.
Backup configuration	.gz	File that contains a backup of the Oracle Enterprise Communications Broker software configuration. You can apply this file to restore a previous configuration if required.
Configuration CSV	.csv	Alternative format for configuration files, allowing management of a subset of the entire configuration for convenience. The configuration CSV can include the entire configuration. Configuration CSV file format is presented below.
Log	Text	Log files that contain information about the various aspects of the Oracle Enterprise Communications Broker. For example, information logged about the ACLI, SIP, or H323. Note: Only the Download and Delete functions are applicable to log files on the Oracle Enterprise Communications Broker.

Note:

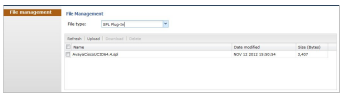
You can activate an LST file or an SPL file dynamically during an upload, if required. You can also immediately apply a backup configuration file during the upload process.

The following illustrations show an example of each file type screen.

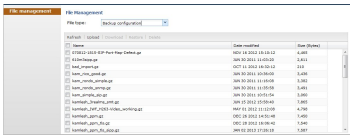
Local Route Table File Management



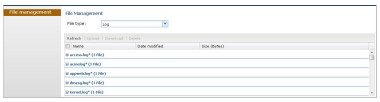
SPL Plugin File Management



Backup Configuration File Management

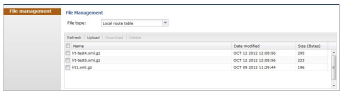


Log File Management



Accessing the System Tab

After logging into the GUI, click the “System” tab. The System files window displays.



This window shows the System files currently stored on the Oracle Enterprise Communications Broker. The “Local route table” files display by default. The following table describes the columns on this page.

Column	Description
File Type	Lists the applicable System files you can select to display in the window. Valid values are: Local subscriber table (LST) SPL Plug-in (SPL) Backup configuration Configuration CSV Log Software image
Name	Name of the file(s) associated with the file type selected. All file names within a group have common file formats (for example, the local route table can consist of a group of files in the format “<filename>.xml”.)
Date Modified	Month, day, year, and time that the file was last modified. Format is: <MM><DD><YYYY><HH><MM><SS>.
Size (Bytes)	Total size of this file (in bytes).

Column	Description
Group Name Note: This column is hidden by default. For more information about hidden columns, see Customizing the Page Display.	Name of the group to which this file belongs. For example, in the screen above, the file called “JayaRoute1.xml”, belongs to the Group Name “JayaRoute1.xml”, and the file called “lst227.xml” belongs to the Group Name “lst227.xml.”

The following table describes the buttons on this page

Button	Description
Refresh	Updates the screen to display the lastest data.
Upload	Uploads a file type from your server or PC to the Oracle Enterprise Communications Broker. The LST, SPL, and backup configuration upload process provide the option of dynamically applying these files to the Oracle Enterprise Communications Broker.
Download	Downloads the file type from the Oracle Enterprise Communications Broker to your local server or PC (typically to the download directory on your system).
Restore (Applicable to the “Backup configuration” file type only.)	Restores and applies a Backup configuration file to the Oracle Enterprise Communications Broker.
Delete	Deletes the file type from the Oracle Enterprise Communications Broker.

Uploading a File

Procedure and conditions around file upload on the Oracle Enterprise Communications Broker.

You can upload any of the following file types from your local server or PC to the Oracle Enterprise Communications Broker:

- Local subscriber table (LST)
- SPL Plug-in (SPL)
- Backup configuration

> **Note:**
>
> You cannot upload log files.

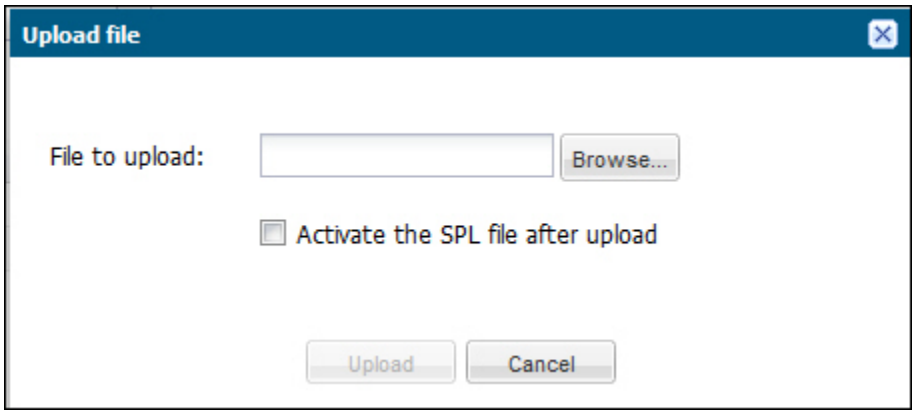
You can dynamically activate the “Local route table” and “SPL Plug-in”during the upload process, if required. You can also immediately restore a backup configuration file after an upload is complete.

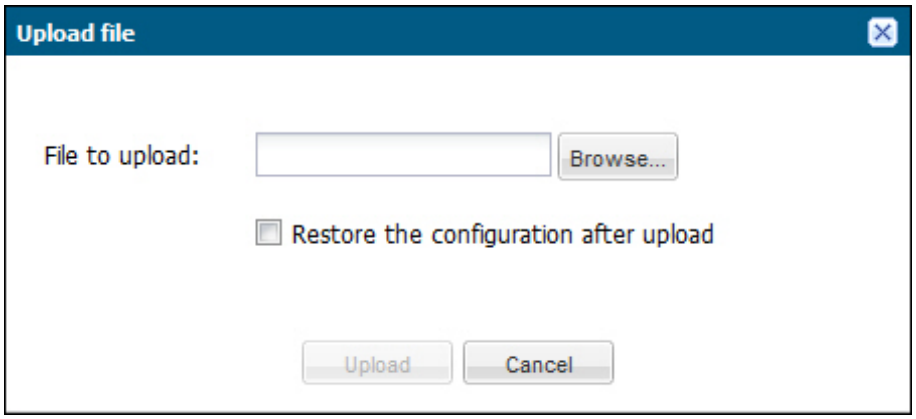
1. (optional) In the “Select the file type” field, select the type of file you want to upload from your local server or PC to the Oracle Enterprise Communications Broker. Valid types of files are:
   - Local subscriber table (LST)

- SPL Plug-in (SPL)
- Backup configuration

> **Note:**
>
> You can click the <Upload> button without selecting a file from the list of files that display

2. In the “Name” column, place a checkmark next to the file you want to upload.
3. Click <Upload>. The following are examples of the dialog box that display, dependant on which file type you chose.





4. In the “File to upload” field, click the <Browse> button, and navigate to the location on your server or PC where the file resides.

> **Note:**
>
> The file extension on the file must be applicable to the file type you select. For example, an SPL Plug-in file must have the file format of “<filename>.lua ”. The following table indicates the file formats required for each File Type, and the applicable directory to which the upload process stores the file on the Oracle Enterprise Communications Broker.

File Type	File Format	Directory
Local subscriber table (LST)	.xml	/code/lst
SPL Plug-on (SPL)	.lua	/code/spl
Backup Configuuration	.gz	/code/bkups

If you select a file with an incorrect file extension, the following message displays: “The file name extension doesn’t match the file type. The file should have the extension: <file type extension>” (For example, “.xml").

5. Perform the following, based on your filetype.

   For the “Local subscriber table” file type, place a checkmark in the “Activate the LST file after upload” box, to immediately apply the LST to the Oracle Enterprise Communications Broker after upload is complete.

   or

   For the “SPL Plug-in” file type, place a checkmark in the “Activate the SPL file after upload” box, to immediately apply the SPL file to the Oracle Enterprise Communications Broker after upload is complete.

   or

   For the “Backup configuration” file, place a checkmark in the “Restore the configuration after upload” box, to immediately apply a previous backed up configuration file to the Oracle Enterprise Communications Broker after upload is complete. Uncheck the box to restore the backup configuration at a later time. You can use the <Restore> button to restore the configuration to the Oracle Enterprise Communications Broker when required.

6. Click <Upload> or click <Cancel> to cancel the upload function.

   After clicking <Upload>, the Oracle Enterprise Communications Broker checks if the file you are uploading already exists on the system. If the file exists, the following prompt displays:

   “Would you like to replace the current file?”

   Click <Yes> to replace the file.

   or

   Click <No> to cancel the upload function.

Downloding a File

Procedure and conditions around file download from the Oracle Enterprise Communications Broker.

You can upload any of the following file types from your local server or PC to the Oracle Enterprise Communications Broker:

- Local subscriber table (LST)
- SPL Plug-in (SPL)
- Backup configuration

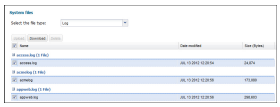
1. In the “Select the file type” field, select the type of file you want to upload from your local server or PC to the Oracle Enterprise Communications Broker. Valid types of files are:
   - Local subscriber table (LST)
   - SPL Plug-in (SPL)

- Backup configuration

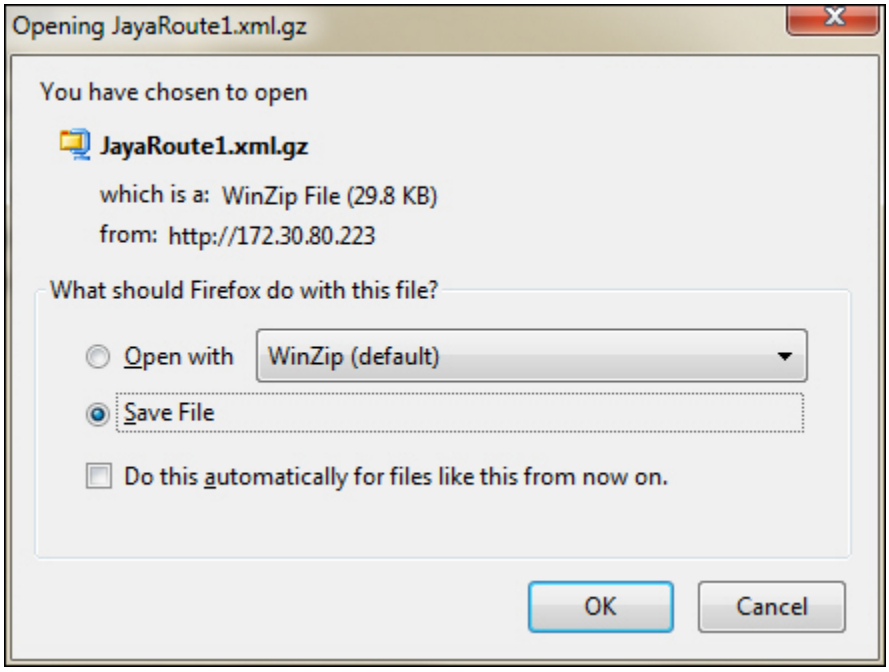
> **Note:**
>
> You can click the <Upload> button without selecting a file from the list of files that display

2. In the “Name” column, place a checkmark next to the file you want to upload.

   For Log file types, you can select multiple log files to download, or place a checkmark in the box to the left of the “Name” column heading to select all log files to download. When downloading multiple log files, the File Management GUI compresses the files into one “.tar” file and downloads that file to your local server or PC.



3. Click <Download>. The following is an example dialog box that displays



4. Click “Open with” and select the application for which to open the file type for decompressing and/or editing. Or click “Save File” to save the file type to your local server or PC.

5. Click <OK>. The file type downloads to the folder on your local server or PC where your Browser sends all downloads (typically your “Download” folder) or opens (decompresses) the file type on your local server or PC (typically in the “Download” folder).

Deleting a File

Procedure and conditions around file delete from the Oracle Enterprise Communications Broker.

You can delete any of the following file types from your local server or PC to the Oracle Enterprise Communications Broker:

- Local subscriber table (LST)
- SPL Plug-in (SPL)
- Backup configuration

> **Note:**
>
> You can select a single or multiple files to delete.

1. In the “Select the file type” field, select the type of file you want to upload from your local server or PC to the Oracle Enterprise Communications Broker. Valid types of files are:
   - Local subscriber table (LST)
   - SPL Plug-in (SPL)
   - Backup configuration
2. In the “Name” column, place a checkmark next to the file(s) you want to delete.

   > **Note:**
   >
   > For Log file types, place a checkmark in the box to the left of the “Name” column heading to select all log files to delete.
3. Click <Delete>. The following message displays.

   “Are you sure you want to delete the file?”
4. Click <Yes> to delete the file(s) from the Oracle Enterprise Communications Broker.

   or

   Click <No> to cancel the delete function.

Back up a File

You can backup a configuration file from the Oracle Enterprise Communications Broker (ECB) to your local server or PC. Backup allows you to save configurations that you can restore to the ECB at a later time.

1. From the Web GUI, click **System**.
2. In the Select the file type field, select Backup configuration.
3. Select one or more configuration files to backup to your server or PC.
4. Click **Backup**.

5. Click **OK** to backup the configuration.

   The system downloads the file to your server or PC, typically into the download directory.

Restore a File

You can restore a backed up configuration file to the Oracle Enterprise Communications Broker (ECB).

1. In the **Select the file type** field, select **Backup configuration**.
2. Select a backup file to restore to the ECB.

> **Note:**
>
> **Restore** activates only when you select a backup file.

3. Click **Restore**.
4. Click **Yes**.

   The system downloads the backup file to the ECB. The ECB re-boots and restores the configuration from the backup file.

Configuration CSV Files

The Comma Separated Values (CSV) file is a text format file supported by spreadsheet applications. You can import a CSV file to the Oracle Enterprise Communications Broker that contains its configuration, or you can export the current configuration on the Oracle Enterprise Communications Broker to the CSV file. Also, you can upload parts of your Oracle Enterprise Communications Broker configuration separately, such as users, dial plans and routes.

In the CSV file format, each row is defined on its own line and each column is separated by a comma.

You can create your own CSV configuration files, but be aware of the following rules for proper formatting.

- Empty lines are ignored.
- If an entry contains a comma, enclose it in quotes to prevent it from being treated as a separator.
- The first non-empty line must be the keyword “object:”, followed by the configuration object name that is being configured (shown below as “sip-interface”).

  ```
  object:sip-interface
  ```

- The second non-empty line must be the parameter names of the object(s) to be configured, each parameter name in its own column. This row defines the “labels” for each column for the subsequent rows. Only the attributes you want defined need to be present. You can specify the parameter names in any order, but the data in subsequent rows must be consistent with the “labels” that you define in this row.

  ```
  state,realm-id,description
  ```

- The third non-empty rows define instances (values) for the configuration object, each instance in its own column. In the following example, the third line defines a new sip-

interface with state “enabled”, realm-id “public”, and description “public SIP interface”. These values are based on the “labels” defined in the second row.

```
enabled,public,public SIP interface
```

- On all subsequent rows, you can define any number of instances.
- The next row with an “object” keyword selects a new configuration object that is based on the previous object. You continue to input the data for this object according to the rules stated above. The following example shows a “sip-port” object added that is related to the sip-interface object.

```
object:sip-port
address,port,transport-protocol
192.168.1.1,5060,UDP
192.168.1.1,5061,TCP
```

- In the previous example, “sip-port” is a sub-object of “sip-interface” that creates new sip-ports from the last sip-interface instance (of realm-id “public”).
- Note that the Description field displays all text as one continuous line, unless you insert line breaks. When you want to insert line breaks in the Description field, for example between sentences that you want displayed on separate lines, do the following:
  - From the GUI, in the Description field of a Configuration object, add Line1 to the end of the line where you want the first break to occur. Add Line2 to the end of the next line where you want a break to occur, and so on.
  - In a .csv configuration file, add \010Line1 to the end of the line where you want the first break to occur. Add \010Line2 to the end of the next line where you want a break to occur, and so on.

To create a CSV file that contains system configuration, do the following:

1. Open an application that supports a CSV file.
2. In the first row, first column, enter “object:” followed by a configuration object you want to import.`object:sip-interface`
3. In the second row, and each in its own column, enter the parameter names of the objects to be configured. state,realm-id,description`state,realm-id,description`
4. In the third row, and each in its own column, enter the instances (values) for the configuration objects.`enabled,public,public SIP interface`
5. In subsequent rows, define additional instances (values), as needed.
6. In the next empty row, first column, enter another object if needed, related to the first object (sip-interface).`object:sip-port`
7. Repeat steps 3 through 5 for this object.
8. Save the file as a .CSV.
9. Upload the configuration file using the upload button from the applicable dialog. (For example, upload a CSV file of users from the User database.)

Caveats surrounding the creation of CSV configurations include:

- Files are written to the volatile directory of the file system on the system. For the Acme Packet 4500, this is the “/ramdrv/” directory. For the appliance and virtual machines, it is the “/var/” directory.
- Import and export occurs to and from the editing configuration.

- All error messages are printed to the screen, where the command was issued. Line numbers are provided with the error when possible.
- Objects and attributes cannot be set to instances (values) that are not allowed. For example, you cannot set an IP address to "enabled". Parsing continues normally after this error.
- If an object cannot be written (i.e. key field is missing), then that object is discarded and parsing continues as normal.
- The import is additive. Each object that is imported is expected to be new to the configuration. If there is already an object with the same key present, it generates an error 409 and is discarded. Parsing continues as normal after the error.

Upgrade Software - Web GUI System Tab

You can upgrade the system software from the System tab on the Web GUI. The system requires a reboot after the upgrade.

1. From the Web GUI, click the `System` tab.
2. Click `Upgrade Software`.
3. Click `Verification`.
4. Verify that system health, synchronization health, current configuration version, and disk usage are appropriate and adequate for the upgrade.
5. From the drop-down list, select `Upload method` , and select one of the following methods.
   - Local. Use to select a file from your system for transfer.
   - Flash. Use to select a file already on the device.
   - Network. Use to specify parameters for network boot by way of file transfer.

   The system displays the Upgrade Software dialog with the fields required for your upgrade.
6. Complete the required fields.
   - Software file to upload. (Local) Use `Browse` to locate the file on your local system.
   - Software file. (Flash) The location and name of the file on the device.
   - Boot file. (Network) The complete name of the boot file.
   - Host IP. (Network) The IP address of the FTP server.
   - FTP username. (Network) The user name to log onto the FTP server.
   - FTP password. (Network) The password to log onto the FTP server.
7. Optional. Select `Reboot after upload`.
8. Click `Complete`.
   - If you did not select `Reboot after upload`, the system displays a message stating that a reboot is required for the changes to take effect.
   - If you selected `Reboot after upload`, the system displays a message stating that it is about to reboot.
9. Click `OK`.

   If you selected `Reboot after upload`, the system reboots.

System Reboot

You can manually reboot the Oracle Enterprise Communications Broker (ECB) from the Web GUI. Note that when you reboot the system from the Web GUI, the Web GUI is unavailable until the reboot is complete. If you have a High Availability (HA) deployment, connectivity to the secondary ECB is lost until the reboot is complete.

When the reboot is complete, the primary and secondary systems both display the logon screen. You must manually log on to each system.

When you perform a reboot from the Web GUI	The system behaves
and no boot is in process and the system is not failing over to the secondary system in an HA environment	The GUI session closes and the system displays the Logon screen. You cannot log on to the Web GUI until the reboot is complete on the ECB.
and a reboot is already in progress	The system displays a message indicating that a reboot cannot occur. The first reboot must complete before another reboot is initiated.
and the primary system is currently failing over to the secondary system in an HA environment	The system displays a message indicating that a reboot cannot occur. The HA switch over is underway. The secondary ECB is updating and getting its configuration from the primary ECB.

Displaying Log Files

The Oracle Enterprise Communications Broker allows the user to view log files without having to download them.

1. Click the **System** tab.

   The Oracle Enterprise Communications Broker displays the system navigation panel to the left of the associated controls.

2. Click the **System** tab's **File management** link.

   The Oracle Enterprise Communications Broker displays the **File Management** dialog, which includes **File type** drop down control.

3. Select the **Log** file type.

   The Oracle Enterprise Communications Broker displays file list, displaying all log file categories.

4. Expand a log file category and select a log file by checking the file's check box.

   The Oracle Enterprise Communications Broker enables all applicable command links on the File Management control bar, including the **View** link.

5. Click the **View** link.

   The Oracle Enterprise Communications Broker displays the **Viewing log:[filename]** dialog with the log file's contents. This dialog includes **Close** and **Refresh** buttons.

Displaying System Health

The Oracle Enterprise Communications Broker provides a widget that allows the user to see your device's current health score and state.

1. Click the **Widgets** tab.

   The Oracle Enterprise Communications Broker displays the widget navigation panel to the left of the associated controls.

2. Find and click the **System health** widget group in the **System** widget category.

   The Oracle Enterprise Communications Broker displays the **System health** widget display types, including the link to the **Table** display.

3. Click the **Table** link.

   The Oracle Enterprise Communications Broker displays the **System health** table.

4. Click the Synchronization health button to show extended details on the system's current status.

   The system displays the popup Synchronization health table. The table's information is useful to determine the system's relative ability to act as primary in an HA configuration. If the system is deployed in an HA configuration, there is also a Switch over log button, which allows the user to display information about HA switchover events.

Obtaining Support Information

The Oracle Enterprise Communications Broker allows the user obtain a pre-defined file containing information that support personnel normally request.

1. Click the **System** tab.

   The Oracle Enterprise Communications Broker displays the system navigation panel to the left of the associated controls.

2. Click the **System** tab's **Support information** link.

   The Oracle Enterprise Communications Broker displays the **Support Information** dialog, which includes the **Support information** button allowing the user to generate support information output.

3. Click the **Support information** button.

   The Oracle Enterprise Communications Broker displays a **Progress** message box, which indicates the system is generating support information output. When complete, your browser displays a dialog allowing you to decide what to do with the support-info.log file.

4. Follow the dialog's instructions to select the application you want to use to display your support-info.log file or save the file locally.

16
Active Directory Modifications

When using the Oracle Enterprise Communications Broker's LDAP configuration to access authentication and/or routing information from Active Directory (AD), the user must prepare AD to serve those functions. For authentication, the user can add an Oracle-supplied DLL to their system to capture password hashes during password changes and store them for authentication. This section describes that preparation.

The Oracle-supplied DLL, **oecbpwdcn.dll**, is an OSD DLL that provides the Windows-specific password hash capture function. When a user changes their password, the DLL intercepts the hash of the password and stores it for SIP authentication. The user's password is never visible in clear-text.

Related AD changes consists of the following, which can be done manually or via Oracle-provided scripts:

1. Create orclDigestRealmAttribute attribute (to store digest realm name) and associate it users.
2. Create orclDigestPwdAttribute attribute (to store hashed password) and associate with users.
3. Create orclAgentNameAttribute and associate it with users.

You can refer to **http://www.microsoft.com/technet/prodtechnol/windows2000serv/technologies/activedirectory/howto/adschema.mspx** for instructions on managing AD. You can manually add the following entries into AD, as follows:

```
dn: cn=orcldigestrealmattribute,cn=schema,cn=configuration,dc=example,dc=com
changetype: add
objectClass: top
objectClass: attributeSchema
cn: orclDigestRealmAttribute
instanceType: 4
attributeID:
1.2.840.113556.1.8000.2554.54362.52699.4250.17878.46369.10622351.7266019.1
attributeSyntax: 2.5.5.4
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: orclDigestRealmAttribute
adminDescription: Oracle ECB Digest Realm
oMSyntax: 20
lDAPDisplayName: orclDigestRealmAttribute
name: orclDigestRealmAttribute
```

This creates the attribute to which **oecbpwdcn.dll** stores password hashes.

```
dn: cn=orcldigestpwdattribute,cn=schema,cn=configuration,dc=example,dc=com
changetype: add
objectClass: top
objectClass: attributeSchema
cn: orclDigestPwdAttribute
instanceType: 4
attributeID:
1.2.840.113556.1.8000.2554.54362.52699.4250.17878.46369.10622351.7266019.2
```

```
attributeSyntax: 2.5.5.4
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: orclDigestPwdAttribute
adminDescription: Oracle ECB Digest Password
oMSyntax: 20
lDAPDisplayName: orclDigestPwdAttribute
name: orclDigestPwdAttribute
```

This creates an attribute that can be used for routing, specifically by providing a field for storing the users' Agent.

```
dn: cn=orclagentnameattribute,cn=schema,cn=configuration,dc=example,dc=com
changetype: add
objectClass: top
objectClass: attributeSchema
cn: orclAgentNameAttribute
instanceType: 4
attributeID:
1.2.840.113556.1.8000.2554.54362.52699.4250.17878.46369.10622351.7266019.3
attributeSyntax: 2.5.5.4
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: orclAgentNameAttribute
adminDescription: Oracle ECB Agent Name
oMSyntax: 20
lDAPDisplayName: orclAgentNameAttribute
name: orclAgentNameAttribute
```

> **Note:**
>
> You must replace %AD_DOMAN_NAME% with your AD domain name, such as dc=acme,dc=com.

For convenience, two LDIF files are provided to facilitate adding these two attributes. They are "addOrclECBAttribute.ldif" and "addUserObjClass.ldif". To add the two attributes automatically:

1. Make sure the Active Directory Schema Snap-In is installed by following the directions from:
   - **http://social.technet.microsoft.com/wiki/contents/articles/20319.how-to-create-a-custom-attribute-in-active-directory.aspx** or
   - **http://technet.microsoft.com/en-us/library/cc759633(v=ws.10).aspx**
2. Open the two files and replace %AD_DOMAN_NAME% with your actual AD domain name, such as dc=acme,dc=com.
3. Run the command "**ldifde –i –f addOrclECBAttribute.ldif –v**" to create the three attributes.
4. Then run the command "**ldifde –i –f addUserObjClass.ldif –v**" to associate the attributes to AD users.
5. Reload the AD schema or reboot AD.
6. Verify that the two attributes are present by checking users to see that attributes are available to them.

In addition to AD schema modification, follow the steps below to install **oecbpwdcn.dll**.

1. Install OID Password Change Notification (oecbpwdcn) DLL, by simply copying the oecbpwdcn.dll to your AD WINDOWS\system32
2. Using regedt32 to change the registry and enable the DLL. Invoke regedt32 and modify the registry setting HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa \Notification Packages. Add "oecbpwdcn" to the end of this list. Example registry entries, including oecbowdcn, could now include:
   - RASSFM
   - KDCSVC
   - WDIGEST
   - scecli
   - oecbpwdcn
3. Reboot AD.

Test your deployment as follows:

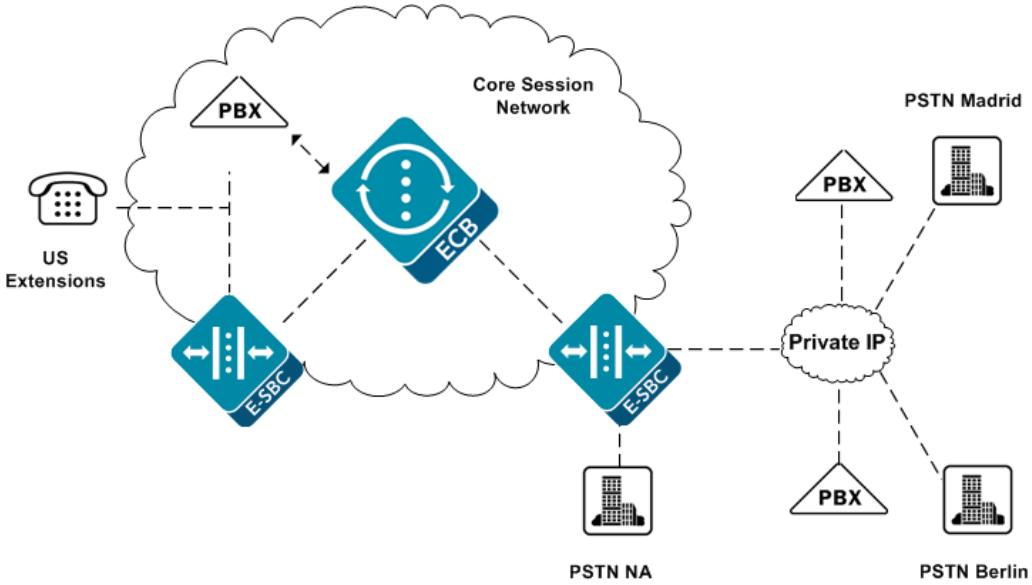
1. Assign a digest realm name to user's orclDigestRealmAttribute in AD. You can use script modifyUsersDigestRealmName.vbs to modify this attribute for all users. Right click on modifyUsersDigestRealmName.vbs and select "Run with Command Prompt"
2. Modify user password for any AD user (or reset the password)
3. Search against AD and look up the AD user and orclDigestPwdAttribute should have the generated hash value.

You can use a script named **displayUsersDigestRealmPassword.vbs** to display the values from all users. To do this, right-click on **displayUsersDigestRealmPassword.vbs** and select "Run with Command Prompt".

17
Configuration Examples

This appendix provides examples for configuring the Oracle Enterprise Communications Broker using sample configurations targeting specific environment models. The Oracle Enterprise Communications Broker is a flexible tool that provides a variety of configuration options that can achieve similar results. This chapter helps you identify a model to which your deployment most closely aligns and presents configurations for meeting common needs. Use the examples within to identify configuration options that track closely to the needs of your specific deployment.

Consider the diagram below. A Oracle Enterprise Communications Broker is deployed at the center of the session network and is managing SIP signaling traffic for an enterprise headquartered in the US, with branch offices in the US, Spain and Germany.



There are multiple E-SBCs, PBXs and tie lines handling session services. The Oracle Enterprise Communications Broker must normalize multiple signaling formats, integrate multiple vendor processes, provision varied session services and handle an enterprise user database.

This section presents configuration settings that accommodate this deployment and exemplify the configuration options that you can use for your deployment.

Configuration Sequence

When first configuring the Oracle Enterprise Communications Broker for service, follow the sequence below to establish objects that you need in ensuing object configurations.

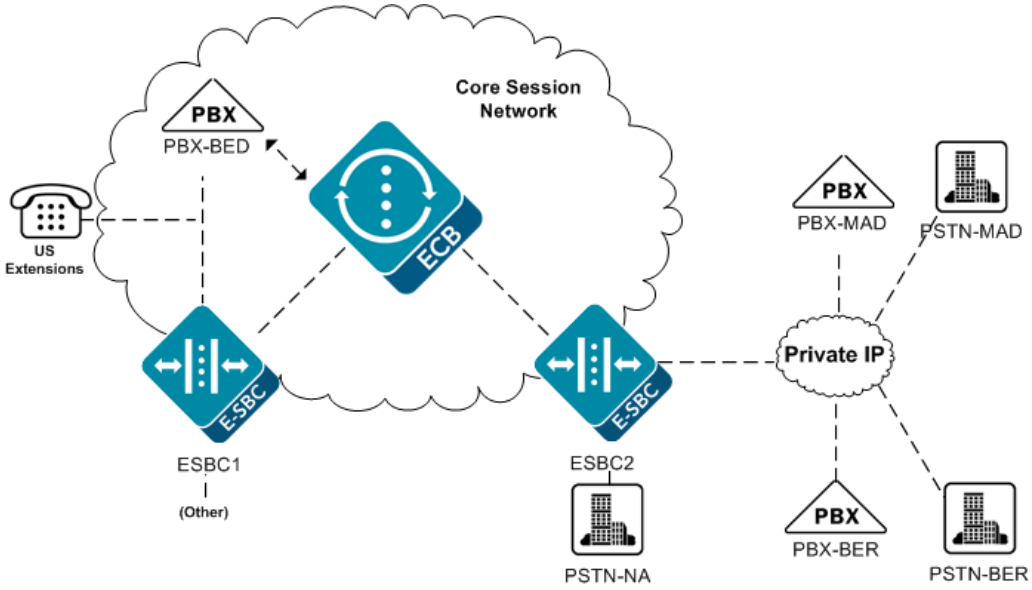
1. Agents - Agents help to segregate your network logically and geographically, providing Agent configuration also specifies the hops available to you on your network for routes. In addition Users often use Agents to help specify their contexts.

2. Dial Plans - Dial Plans specify all the contexts that you use in routes. Users may also be configured with contexts.
3. Users - It is often best to configure Users with existing Agents and Dial Plans to specify their locations and applicable policies.
4. Routes - Routes often use Agents to specify source and destinations.

You will find this sequence is of decreasing value over time. In addition, it is not required that you follow this sequence at any point. The flexibility of the Oracle Enterprise Communications Broker's configuration options allow you perform steps such as creating new contexts as you create users. The sequence is most useful for understanding how to piece together the elements of the Oracle Enterprise Communications Broker.

Initial Agent Configuration

The diagram below extends upon the previous diagram adding specific names for agents that are required for this Oracle Enterprise Communications Broker deployment.



The table below presents Agent configurations for those on the diagram. This table is also intended to provide examples from which you can glean guidelines for configuring your own agents.

Configure each agent, listed below, with IP address, Port and Transport mode. In addition, all Agents take the default Number Translation Mode, which is e.164.

Hostname	Source Context	# Translation Mode	# of Digits	Prepend Prefix
PBX-BED		e.164		
PBX-MAD		e.164		
PBX-BER		e.164		
ESBC1		e.164		
ESBC2		e.164		
PSTN-NA		e.164		

Hostname	Source Context	# Translation Mode	# of Digits	Prepend Prefix
PSTN-MAD		e.164		
PSTN-BER		e.164		

No other configuration is necessary to support the early examples in this appendix. Additional agent configuration is added when necessary.

Dial Plan Strategies

Oracle Enterprise Communications Broker configuration design provides you with the flexibility to make the same settings, such as country code, on multiple elements. To the extent that child elements inherit properties from parent elements, endeavor to elegantly cover the basic requirements of your deployment with your initial configurations while preserving configuration options in child objects to meet the needs of exceptions and expansion.

The simplest way to approach dial plan configuration is to base your corporate contexts on your enterprise's branch offices. You configure dials plan and patterns comes within those contexts. The parent context establishes rules that you need enforced across the enterprise. Each branch office gets a child context that inherits from both the corporate parent and the rules associated with the country in which branch resides (geographic context).

Recall that the Oracle Enterprise Communications Broker comes pre-configured with the vast majority of geographic contexts you need. These contexts include the country code. By setting a geographic location to your branch office, you inherit country code. These contexts also include a description (of their location), which has no relevance to signaling processing.

The following corporate context configurations apply to the early enterprise configuration models presented in the ensuing sections.

Name	Geographic location	Country Code	Outside line prefix
acme			
acme.bedford	NA		9
acme.madrid	EU.Spain		
acme.berlin	EU.Germany		

This appendix uses dial plans and dial patterns to establish differentiation between configuration models, with all using the same routes. For this reason, this appendix strays from the suggested configuration sequence and sets up routes next.

Route Strategies

Route configuration consists of mapping out an extensible strategy according to your deployment model and connecting agents together. Configure all agents as simply as possible and create only as many routes as are necessary. Careful planning allows you to create routes that serve multiple purposes simultaneously.

You may recall that initial configuration procedures has user configuration preceding route configuration. This configurations design, however, covers all expected users without specific user configuration.

Routes Configured to Support both Small and Medium/Large Enterprise Models

Route #	Source Agent	Calling #	Called #	Dest Agent	Route	Cost
#1	*	*	34*	*	PSTN-MAD	20
#2	*	*	*	PSTN-MAD	ESBC2	0
#3	*	*	*	PBX-MAD	ESBC2	0
#4	*	*	49*	*	PSTN-BER	20
#5	*	*	*	PSTN-BER	ESBC2	0
#6	*	*	*	PBX-BER	ESBC2	0
#7	*	*	1*	*	PSTN-NA	20
#8	*	*	*	*	PSTN-NA	70

The table below explains the purpose of each route in the table above.

Route #	Description
#1	For traffic destined to Spain preceded with a "34" (Spain Country Code) and sourced anywhere, send to the PSTN agent in Madrid.
#2	For traffic destined to the PSTN in Madrid, use ESBC2.
#3	For traffic destined within the enterprise in Madrid, use ESBC2.
#4	For traffic destined to Germany preceded with a "49" (German Country Code) and sourced anywhere, send to the PSTN agent in Berlin.
#5	For traffic destined to the PSTN in Berlin, use ESBC2.
#6	For traffic destined within the enterprise in Berlin, use ESBC2.
#7	For traffic destined anywhere preceded with a "1" (US Country Code) and sourced anywhere, send to the North American PSTN agent.
#8	Default Route - If unable to determine any preferable route, use the most expensive route, which offloads traffic to the North American PSTN agent. Note the cost of 70, which is the highest cumulative cost of any other route set.

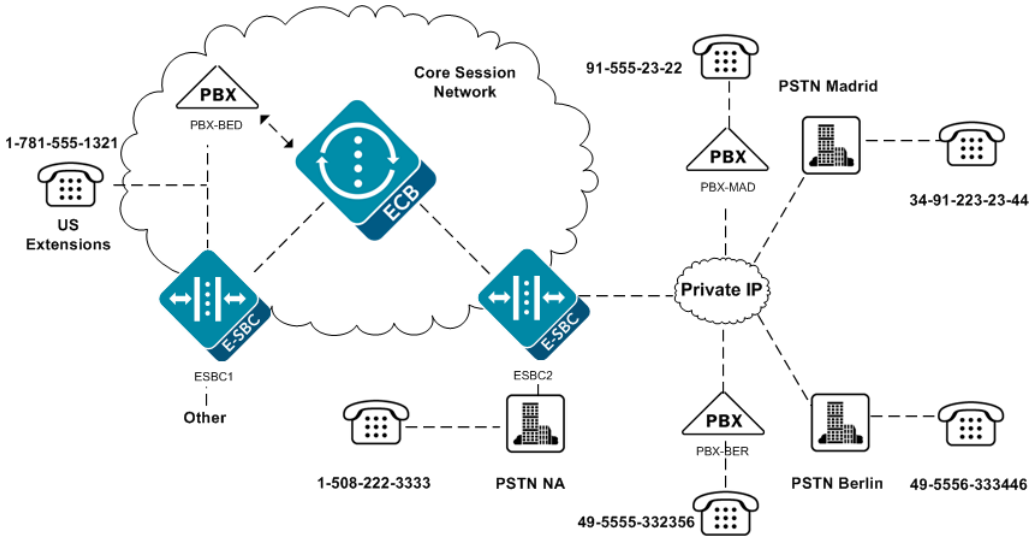
Note:

Routes 1, 4 and 7 are examples of tail hop routing, which keeps traffic within the enterprise network for as long as possible before issuing a call to Germany locally in Berlin.

Small Enterprise Model - v2

The tables below present the dial plan for a small enterprise. Recall that the key characteristic for this model is the absence of overlap of dial patterns across branch locations (contexts).

The diagram below adds example extension numbers for the small enterprise model that must be reachable.



Dial Patterns Configured for the acme Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	1xxx	1	781555		
	2xxx	34	91555		
	3xxxxx	49	5555		

Applicable user database entries

Number	Dialing Context	Agent
+17815551xxx		PBX-BED
+34915552xxx		PBX-MAD
+4955553xxxxx		PBX-BER

Recall the configuration of the child contexts. These contexts inherit the rules of the corporate context, acme, and are also configured to inherit the geographic location contexts for the countries in which they reside.

Recall also the routes configured. Numbers preceded with the appropriate country code that do not match these dial patterns go to the PSTN of their respective countries.

Reviewing the fields that do not require configuration is equally instructive to the user. Again, based on the inheritance of rules from parent contexts, a means of basic access is available throughout the enterprise. In addition, based on the higher precedence of child context configurations, these fields are available to you to configure for special requirements at the separate branches.

This configuration provides the results presented in the table below.

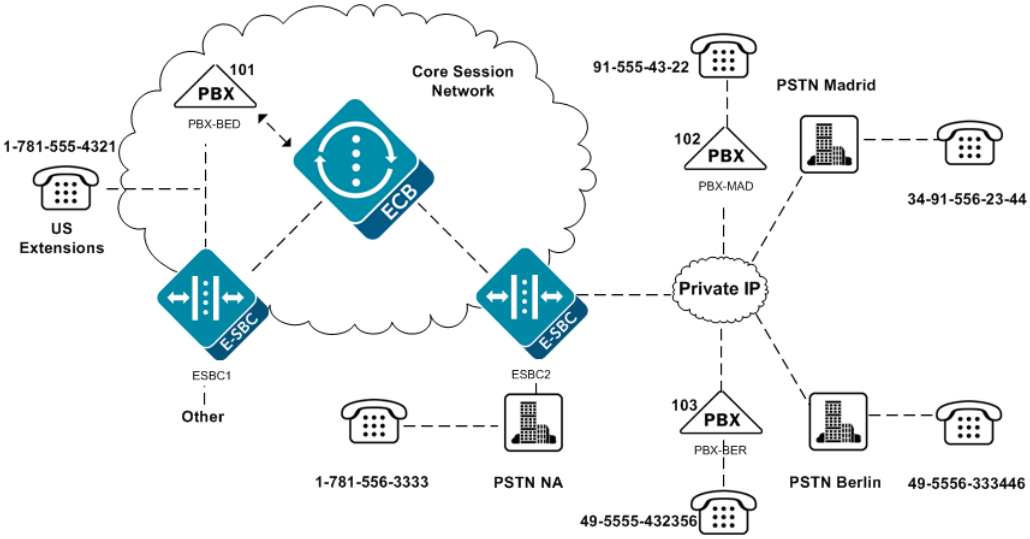
From	Dial String	Transformation	Result
acme	1321	+17815551321	Call is directed to PBX-BED as e.164
acme	2322	+345552322	Call is directed to PBX-MAD as e.164

From	Dial String	Transformation	Result
acme	332356	+495555332356	Call is directed to PBX-BER as e.164

Large Enterprise Model - v2

The following tables present the dial plan for a Medium/Large enterprise. Recall that the key characteristic for this model is the overlap of dial patterns for each branch location (context).

The diagram below adds example extension numbers for the medium to large enterprise model that must be reachable.



In this case, you can configure specific dial patterns for the acme context that each child context inherits. These dial patterns provide the Oracle Enterprise Communications Broker with the means of distinguishing (disambiguation) between branches, even though the extension patterns still begin with the same number.

Dial Patterns Configured for the `acme` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
101					acme.bedford
102					acme.madrid
103					acme.berlin
555				helpdesk@acme.com	

Each child context inherits these dial patterns from the acme context, providing users with a means of hierarchically assigning context.

In contrast to the small enterprise model, note the use of dial patterns specific to each branch. These contexts work in conjunction with the parent context's parents, delivering signaling with the enterprise's own prefixes to the branch's PBX.

Dial Patterns Configured for the `acme.bedford` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	4xxx		781555		

Dial Patterns Configured for the `acme.madrid` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	4xxx		91555		

Dial Patterns Configured for the `acme.berlin` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	4xxxxx		5555		
111			berlinhelpdesk@acme.com		

Applicable user database entries

Number	Dialing Context	Agent
+17815554xxx		PBX-BED
+34915554xxx		PBX-MAD
+4955554xxxxx		PBX-BER

In contrast to the small enterprise model, note the use of dial patterns specific to each branch.

This configuration provides the results presented in the table below.

From	Dial String	Transformation	Result
acme.bedford	4321	+17815554321	Call is directed to PBX-BED as e.164
acme.bedford	1024322	+34915554322	Call is directed to PBX-MAD as e.164
acme.bedford	103432356	+495555432356	Call is directed to PBX-BER as e.164
acme.madrid	4322	+34915554322	Call is directed to PBX-MAD as e.164
acme.madrid	1014321	+17815554321	Call is directed to PBX-BED as e.164
acme.madrid	103432356	+495555432356	Call is directed to PBX-BER as e.164
acme.berlin	432356	+495555432356	Call is directed to PBX-BER as e.164
acme.berlin	1014321	+17815554321	Call is directed to PBX-BED as e.164
acme.berlin	1024322	+34915554322	Call is directed to PBX-MAD as e.164

Emergency Dial Configurations

To dial emergency numbers, you add dial patterns to the corporate context child context for the emergency numbers. Configure these dial patterns with a Replace URI transformation that inserts the RFC 5031 compliant URN for emergency services.

Dial Patterns Configured for the `acme.bedford` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	911			URN:service:sos	

Dial Patterns Configured for the `acme.madrid` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	112			URN:service:sos	

Dial Patterns Configured for the `acme.berlin` Context

Prefix	Pattern	Country Code	Replace Prefix	Replace URI	Go To Context
	112			URN:service:sos	

In addition to the dial patterns above, the system needs three new routes.

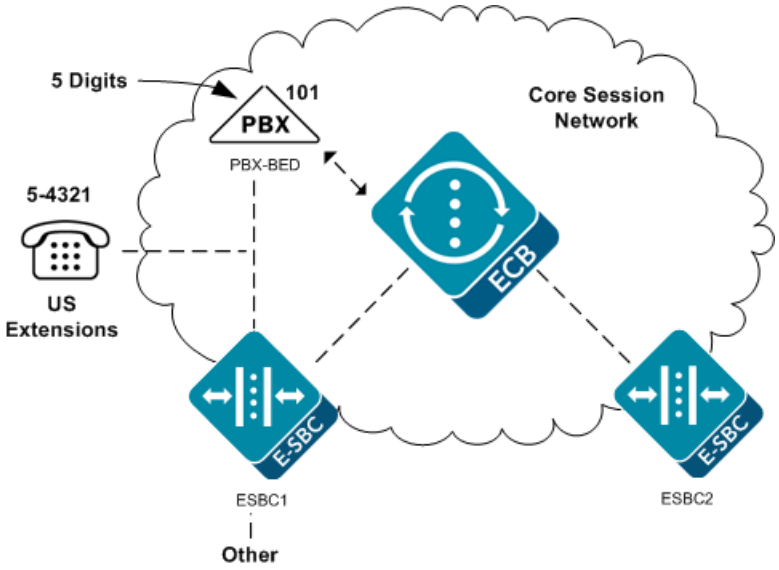
Route #	Source Agent	Calling #	Called #	Dest Agent	Route	Cost
#1	*	1*	"service:sos"	*	PSTN-NA	0
#2	*	34*	"service:sos"	*	PSTN-MAD	0
#3	*	49*	"service:sos"	*	PSTN-BER	0

This configuration provides the results presented in the table below.

From	Dial String	Transformation	Result
acme.bedford	911	URN:service:sos	Call is directed to PSTN-NA as emergency URN
acme.madrid	112	URN:service:sos	Call is directed to PSTN-MAD as emergency URN
acme.berlin	112	URN:service:sos	Call is directed to PSTN-BER as emergency URN

Alternate Translation Modes

As described in agent configuration instructions, a translation mode specifies the format required by that agent. The configuration normally applies to a PBX when it is the last agent in the path to the user.



This configuration implements a different configuration for PBX-BED, as follows.

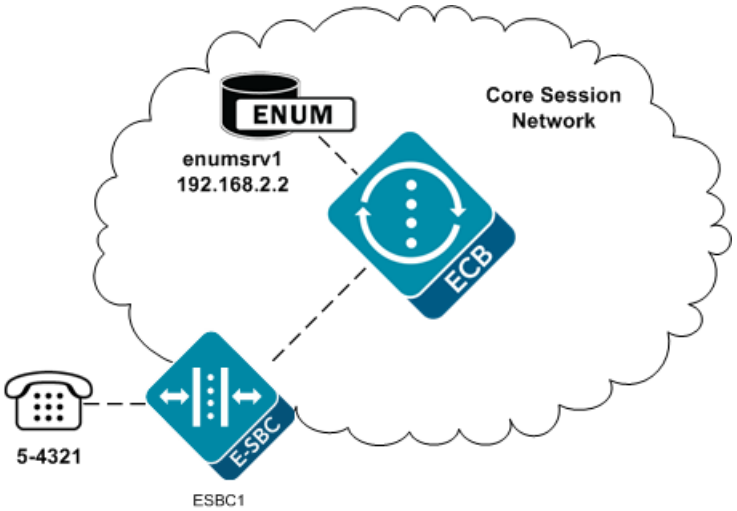
Hostname	Source Context	# Translation Mode	# of Digits	Prepend Prefix
PBX-BED		n-digit-dialing	5	

This configuration produces the following results.

From	Dial String	Transformation	Result
acme.bedford	54321	+17815554321	Call is directed to PBX-BED as a 5 digit number (54321)
acme.madrid	10154321	+17815554321	Call is directed to PBX-BED as a 5 digit number (54321)
acme.berlin	10154321	+17815554321	Call is directed to PBX-BED as a 5 digit number (54321)

ENUM Example Configuration

This example provides for the requirement that the signaling make an ENUM dip to resolve some element of the overall signaling path. The configuration must be able to provide ENUM resolutions to provide for the deployment shown below.



This configuration assumes the agent ESBC1 does not have a configured IP address. In addition, all numbers beginning with 5 are routed to ESBC1. To provide resolution for ESBC1, create an ENUM configuration as follows. In this case, the ENUM configuration includes only one server.

Hostname	Domain	Servers	Number Trans Mode	Number of Digits
enumsrv1	acme.com	192.168.2.2		

Next, configure a route for all 5-digit extensions beginning with a 5 to the agent named ENUM.

Route #	Source Agent	Calling number	Called number	Dest Agent	Route	Cost
#1	*	*	5xxxx	*	enum:enumsrv1	

The table below explains the purpose of the route in the table above.

Route #	Description
#1	For traffic destined to a five-digit number beginning with 5, go to ENUM server to resolve the address of ESBC1.

18

Format of Exported Text Files

This section provides a sample and format of each type of exported file from the Web-based GUI. Sample information in these files are provided as a reference for your convenience.

Exported file examples include:

- Session Summary exported file (text format)
- Session Details exported file (text format)
- Ladder Diagram exported file (HTML format)

> **Note:**
>
> Oracle recommends you open an exported text file using an application that provides advanced text formatting to make it easier to read.

Exporting Files

The Web-based GUI allows you to export Monitor and Trace information to a text file from the Sessions, Registrations, Subscriptions and Notable Events Reports, as well as from a specific ladder diagram, or from a page containing the results of a search. The data exports to a file that you can open and view as required.

You can export any of the following to a file:

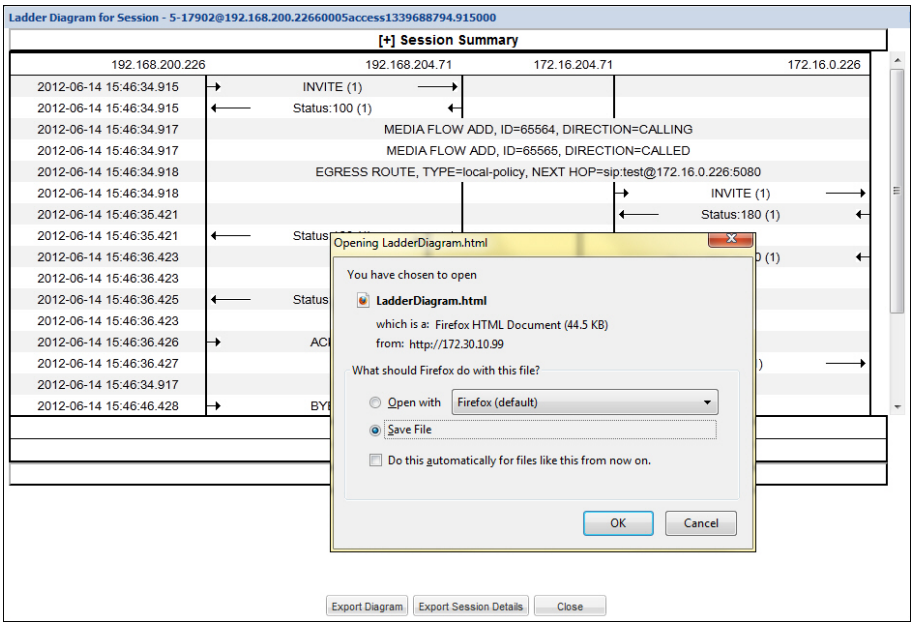
From the Sessions, Registrations, Subscriptions, and Notable Events Reports:

- **Export session details** - Exports the SIP messages and media events associated with the selected session, to a text file.
- **Export summary** - Exports all logged session summary records, to a text file. (Exports ALL call session summary records or the records that matched a search criteria).

From the Ladder Diagram:

- **Export diagram** - Exports all of the information in the Ladder Diagram to an HTML file (Session Summary, SIP Message Details, and QoS statistics).
- **Export session details** - Exports detailed information about the SIP messages and media events in the Ladder Diagram associated with the selected session, to a text file.

The following example shows the export of a Ladder Diagram to a file called LadderDiagram.html.



Session Summary Exported Text File

The following is an example of a Session Summary exported text file from the Web-based GUI.

Example

```
----------Session Summary----------

Startup Time: 2011-09-20 12:58:44.375

State: TERMINATED-200
Duration: 5
From URI: sipp &lt;sip:sipp@172.16.34.16:5060&gt;;tag=1
To URI: sut &lt;sip:service@172.16.34.225:5060&gt;;tag=13451
Ingress Src Address: 172.16.34.16
Igress Src Port: 5060
Igress Dest Address: 172.16.34.225
Igress Dest Port: 5060
Egress Source Address: 192.168.34.225
Egress Source Port: 5060
Egress Destination Address: 192.168.34.17
Egress Destination Port: 5060
Igress Realm: access
Egress Realm: backbone
Igress NetworkIf: access
Egress NetworkIf: backbone

----------Session Summary----------
Startup Time: 2011-09-20 12:58:05.340
State: TERMINATED-200
Duration: 5
From URI: sipp &lt;sip:sipp@172.16.34.16:5060&gt;;tag=1
To URI: sut &lt;sip:service@172.16.34.225:5060&gt;;tag=13450
Ingress Src Address: 172.16.34.16
Igress Src Port: 5060
```

```
Igress Dest Address: 172.16.34.225
Igress Dest Port: 5060
Egress Source Address: 192.168.34.225
Egress Source Port: 5060
Egress Destination Address: 192.168.34.17
Egress Destination Port: 5060
Igress Realm: access
Egress Realm: backbone
Igress NetworkIf: access
Egress NetworkIf: backbone
```

Session Details Exported Text File

The following is an example of the a Session Details exported text file from the Web-based GUI.

Example

```
Session Details:

----------------------------------------
Nov 3 08:50:56.852 On [2:0]172.16.34.225:5060 received from 172.16.34.16:5060

INVITE sip:service@172.16.34.225:5060 SIP/2.0
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: sip:sipp@172.16.34.16:5060
Max-Forwards: 70
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 135

v=0
o=user1 53655765 2353687637 IN IP4 172.16.34.16
s=-
c=IN IP4 172.16.34.16
t=0 0
m=audio 6000 RTP/AVP 0
a=rtpmap:0 PCMU/8000

----------------------------------------
Nov 3 08:50:56.855 On [2:0]172.16.34.225:5060 sent to 172.16.34.16:5060

SIP/2.0 100 Trying
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE

----MBCD Evt
Nov 3 08:50:56.862 On 127.0.0.1:2945 sent to 127.0.0.1:2944
mbcdEvent=FLOW ADD
FlowCollapsed=enabled
FlowDirection=CALLING
FlowID=65541
```

```
MediaFormat=0
MediaReleased=disabled
MediaType=audio/PCMU/
OtherFlowID=0
TOSBits=0
InputSourcev4Addr=
InputSourcePort=0
InputDestv4Addr=172.16.34.225
InputDestPort=10004
OutputSourcev4Addr=192.168.34.225
OutputDestv4Addr=
OutputDestPort=0
InputRealm=access
OutputRealm=backbone
----MBCD Evt
Nov 3 08:50:56.862 On 127.0.0.1:2945 received from 127.0.0.1:2944

mbcdEvent=FLOW ADD
FlowCollapsed=enabled
FlowDirection=CALLED
FlowID=65542
MediaFormat=0
MediaReleased=disabled
MediaType=audio/PCMU/
OtherFlowID=0
TOSBits=0
InputSourcev4Addr=
InputSourcePort=0
InputDestv4Addr=192.168.34.225
InputDestPort=20004
OutputSourcev4Addr=172.16.34.225
OutputDestv4Addr=172.16.34.16
OutputDestPort=6000
InputRealm=backbone
OutputRealm=access

----------------------------------------
Nov 3 08:50:56.865 On [1:0]192.168.34.225:5060 sent to 192.168.34.17:5060

INVITE sip:service@192.168.34.17:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK4od0io20183g8ssv32f1.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.34.225:5060;transport=udp>
Max-Forwards: 69
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 140

v=0
o=user1 53655765 2353687637 IN IP4 192.168.34.225
s=-
c=IN IP4 192.168.34.225
t=0 0
m=audio 20004 RTP/AVP 0
a=rtpmap:0 PCMU/8000

----------------------------------------
Nov 3 08:50:56.868 On [1:0]192.168.34.225:5060 received from 192.168.34.17:5060
```

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK4od0io20183g8ssv32f1.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: <sip:192.168.34.17:5060;transport=UDP>
Content-Length: 0

---------------------------------------
Nov 3 08:50:56.872 On [2:0]172.16.34.225:5060 sent to 172.16.34.16:5060

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: <sip:service@172.16.34.225:5060;transport=udp>
Content-Length: 0

---------------------------------------
Nov 3 08:50:56.872 On [1:0]192.168.34.225:5060 received from 192.168.34.17:5060
SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK4od0io20183g8ssv32f1.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: <sip:192.168.34.17:5060;transport=UDP>
Content-Type: application/sdp
Content-Length: 137
v=0
o=user1 53655765 2353687637 IN IP4 192.168.34.17
s=-
c=IN IP4 192.168.34.17
t=0 0
m=audio 6000 RTP/AVP 0
a=rtpmap:0 PCMU/8000

----MBCD Evt
Nov 3 08:50:56.878 On 127.0.0.1:2945 sent to 127.0.0.1:2944

mbcdEvent=FLOW MODIFY
FlowCollapsed=enabled
FlowDirection=CALLING
FlowID=65541
MediaFormat=0
MediaReleased=disabled
MediaType=audio/PCMU/
OtherFlowID=0
TOSBits=0
InputSourcev4Addr=
InputSourcePort=0
InputDestv4Addr=172.16.34.225
InputDestPort=10004
OutputSourcev4Addr=192.168.34.225
OutputDestv4Addr=192.168.34.17
OutputDestPort=6000
InputRealm=access
```

```
OutputRealm=backbone

---------------------------------------
Nov 3 08:50:56.881 On [2:0]172.16.34.225:5060 sent to 172.16.34.16:5060

SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 INVITE
Contact: <sip:service@172.16.34.225:5060;transport=udp>
Content-Type: application/sdp
Content-Length: 138
v=0
o=user1 53655765 2353687637 IN IP4 172.16.34.225
s=-
c=IN IP4 172.16.34.225
t=0 0
m=audio 10004 RTP/AVP 0
a=rtpmap:0 PCMU/8000

---------------------------------------
Nov 3 08:50:56.883 On [2:0]172.16.34.225:5060 received from 172.16.34.16:5060

ACK sip:service@172.16.34.225:5060 SIP/2.0
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-5
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 ACK
Contact: sip:sipp@172.16.34.16:5060
Max-Forwards: 70
Subject: Performance Test
Content-Length: 0

---------------------------------------
Nov 3 08:50:56.887 On [1:0]192.168.34.225:5060 sent to 192.168.34.17:5060

ACK sip:192.168.34.17:5060;transport=UDP SIP/2.0
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK48k3k1301ot00ssvf1v0.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 1 ACK
Contact: <sip:sipp@192.168.34.225:5060;transport=udp>
Max-Forwards: 69
Subject: Performance Test
Content-Length: 0

---------------------------------------
Nov 3 08:51:01.883 On [2:0]172.16.34.225:5060 received from 172.16.34.16:5060
BYE sip:service@172.16.34.225:5060 SIP/2.0
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-7
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 2 BYE
Contact: sip:sipp@172.16.34.16:5060
Max-Forwards: 70
Subject: Performance Test
```

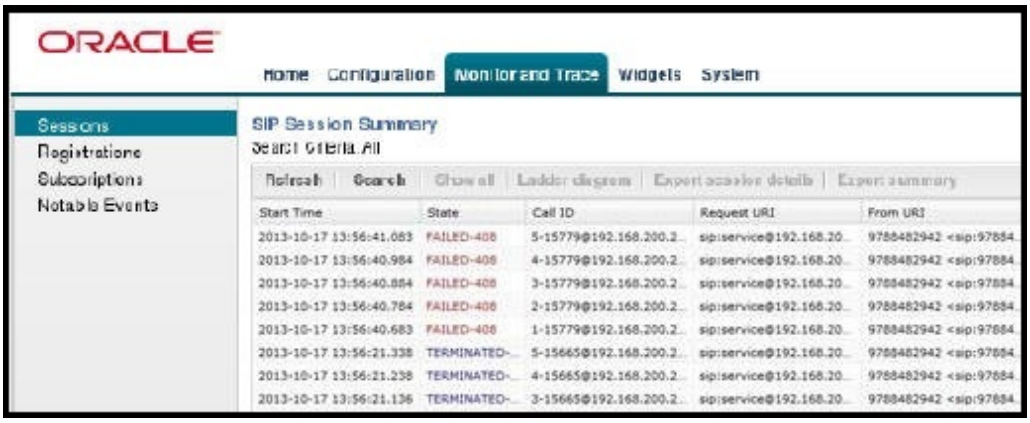
```
Content-Length: 0

---------------------------------------
Nov 3 08:51:01.887 On [1:0]192.168.34.225:5060 sent to 192.168.34.17:5060

BYE sip:192.168.34.17:5060;transport=UDP SIP/2.0
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK5oq46d301gv0dus227f1.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 2 BYE
Contact: <sip:sipp@192.168.34.225:5060;transport=udp>
Max-Forwards: 69
Subject: Performance Test
Content-Length: 0

---------------------------------------
Nov 3 08:51:01.889 On [1:0]192.168.34.225:5060 received from 192.168.34.17:5060

SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.168.34.225:5060;branch=z9hG4bK5oq46d301gv0dus227f1.1
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 2 BYE
Contact: <sip:192.168.34.17:5060;transport=UDP>
Content-Length: 0

---------------------------------------
Nov 3 08:51:01.892 On [2:0]172.16.34.225:5060 sent to 172.16.34.16:5060
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.16.34.16:5060;branch=z9hG4bK-1-7
From: sipp <sip:sipp@172.16.34.16:5060>;tag=1
To: sut <sip:service@172.16.34.225:5060>;tag=2578
Call-ID: 1-668@172.16.34.16
CSeq: 2 BYE
Contact: <sip:service@172.16.34.225:5060;transport=udp>
Content-Length: 0

----MBCD Evt
Nov 3 08:51:01.891 On 127.0.0.1:2945 sent to 127.0.0.1:2944

mbcdEvent=FLOW DELETE
FlowCollapsed=enabled
FlowDirection=CALLING
FlowID=65541
MediaFormat=0
MediaReleased=disabled
MediaType=audio/PCMU/
OtherFlowID=0
TOSBits=0
InputSourcev4Addr=
InputSourcePort=0
InputDestv4Addr=172.16.34.225
InputDestPort=10004
OutputSourcev4Addr=192.168.34.225
OutputDestv4Addr=192.168.34.17
OutputDestPort=6000
InputRealm=access
OutputRealm=backbone
```

```
----MBCD Evt
mbcdEvent=FLOW DELETE
FlowCollapsed=enabled
FlowDirection=CALLED
FlowID=65542
MediaFormat=0
MediaReleased=disabled
MediaType=audio/PCMU/
OtherFlowID=65541
TOSBits=0
InputSourcev4Addr=
InputSourcePort=0
InputDestv4Addr=192.168.34.225
InputDestPort=20004
OutputSourcev4Addr=172.16.34.225
OutputDestv4Addr=172.16.34.16
OutputDestPort=6000
InputRealm=backbone
OutputRealm=access

----------Session Summary----------
Startup Time: 2012-01-25 10:28:30.394

State: TERMINATED-200
Duration: 5
From URI: sipp &lt;sip:sipp@172.16.34.16:5060&gt;;tag=1
To URI: sut &lt;sip:service@172.16.34.225:5060&gt;;tag=2578
Ingress Src Address: 172.16.34.16
Igress Src Port: 5060
Igress Dest Address: 172.16.34.225
Igress Dest Port: 5060
Egress Source Address: 192.168.34.225
Egress Source Port: 5060
Egress Destination Address: 192.168.34.17
Egress Destination Port: 5060
Igress Realm: access
Egress Realm: backbone
Igress NetworkIf: access
Egress NetworkIf: backbone
```

Ladder Diagram Exported HTML File

The following is an example of a Ladder Diagram for a session, exported to an HTML file from the Web-based GUI.

Example



[-] Session Summary

State	TERMINATED-200	Duration	10
From URI	"+2273636" <tel:781-414-2345>;tag=60005	To URI	sut <sip:kam@192.168.204.71:5060>;tag=50004
Ingress Src IP:Port	192.168.200.226:5070	Egress Src IP:Port	172.16.204.71:5060
Ingress Dest IP:Port	192.168.204.71:5060	Egress Dest IP:Port	172.16.0.226:5070
Ingress Realm	access	Egress Realm	core
Ingress Network Intf	M00	Egress Network Intf	M10
Ingress Transport	UDP	Egress Transport	UDP

192.168.200.226	192.168.204.71	172.16.204.71	172.16.0.226
2012-06-14 15:46:34.915	INVITE (1)		
2012-06-14 15:46:34.915	Status:100 (1)		
2012-06-14 15:46:34.917	MEDIA FLOW ADD, ID=65564, DIRECTION=CALLING		
2012-06-14 15:46:34.917	MEDIA FLOW ADD, ID=65565, DIRECTION=CALLED		
2012-06-14 15:46:34.918	EGRESS ROUTE, TYPE=local-policy, NEXT HOP=sip:test@172.16.0.226:5080		
2012-06-14 15:46:34.918		INVITE (1)	
2012-06-14 15:46:35.421		Status:180 (1)	
2012-06-14 15:46:35.421	Status:180 (1)		
2012-06-14 15:46:36.423		Status:200 (1)	
2012-06-14 15:46:36.423	MEDIA FLOW MODIFY, ID=65564, DIRECTION=CALLING		
2012-06-14 15:46:36.425	Status:200 (1)		
2012-06-14 15:46:36.423	MEDIA FLOW LATCH, ID=65564, DIRECTION=CALLING		
2012-06-14 15:46:36.426	ACK (1)		
2012-06-14 15:46:36.427		ACK (1)	
2012-06-14 15:46:34.917	MEDIA FLOW LATCH, ID=65565, DIRECTION=CALLED		
2012-06-14 15:46:46.428	BYE (2)		
2012-06-14 15:46:46.428		BYE (2)	
2012-06-14 15:46:46.430		Status:200 (2)	
2012-06-14 15:46:46.431	Status:200 (2)		
2012-06-14 15:46:46.430	MEDIA FLOW DELETE, ID=65564, DIRECTION=CALLING		
2012-06-14 15:46:46.430	MEDIA FLOW DELETE, ID=65565, DIRECTION=CALLED		

SIP Message Details

[-] QoS Stats

		Total Pkts	Total Octets	RTCP					RTP			QoE	
Flow ID	Direction	Received	Received	Pkt Lost	Avg Jitter	Max Jitter	Avg Latency	Max Latency	Pkt Lost	Avg Jitter	Max Jitter	R-Factor	MOS
65564	CALLING	0	0	0	0	0	0	0	0	0	0	0	0
65565	CALLED	0	0	0	0	0	0	0	0	0	0	0	0

19

Header Manipulation

SIP HMR (Header Manipulation Rules)

SIP header manipulation can also be configured in a way that makes it possible to manipulate the headers in SIP messages both statically and dynamically. Using this feature, you can edit response headers or the Request-URI in a request, and change the status code or reason phrase in SIP responses.

Static SIP Header and Parameter Manipulation allows you to set up rules in your Oracle Enterprise Communications Broker configuration that remove and/or replace designated portions of specified SIP headers. SIP HMR allows you to set up dynamic header manipulation rules, meaning that the Oracle Enterprise Communications Broker has complete control over alterations to the header value. More specifically:

- The Oracle Enterprise Communications Broker can search header for dynamic content or patterns with the header value. It can search, for example, for all User parts of a URI that begin with 617 and end with 5555 (e.g., 617...5555).
- The Oracle Enterprise Communications Broker can manipulate any part of a patterns match with any part of a SIP header. For example, 617 123 5555 can become 617 231 5555 or 508 123 0000, or any combination of those.

To provide dynamic header manipulation, the Oracle Enterprise Communications Broker uses regular expressions to provide a high degree of flexibility for this feature. This allows you to search a specific URI when you do not know that value of the parameter, but want to use the matched parameter value as the header value. It also allows you to preserve matched sections of a pattern, and change what you want to change.

You can apply header manipulation to session agents, SIP interfaces, and realms. You do so by first setting up header manipulations rules, and then applying them in the configurations where they are needed. Within the header manipulation rules, there are sets of element rules that designate the actions that need to be performed on a given header.

Each header rule and each element rule (HMR) have a set of parameters that you configure to identify the header parts to be manipulated, and in what way the Oracle Enterprise Communications Broker is to manipulate them. These parameters are explained in detail, but the parameter that can take regular expression values is **match-value**. This is where you set groupings that you want to store, match against, and manipulate.

Generally, you set a header rule that will store what you want to match, and then you create subsequent rules that operate on this stored value. Because header rules and element rules are applied sequentially, it is key to note that a given rule performs its operations on the results of all the rules that you have entered before it. For example, if you want to delete a portion of a SIP header, you would create Rule 1 that stores the value for the purpose of matching, and then create Rule 2 that would delete the portion of the header you want removed. This prevents removing data that might be used in the other header rules.

Given that you are using regular expression in this type of configuration, this tightly sequential application of rules means that you must be aware of the results to be yielded from the application of the regular expressions you enter. When you set a regular expression match value

for the first rule that you enter, the Oracle Enterprise Communications Broker takes the results of that match, and then a second rule might exist that tells the Oracle Enterprise Communications Broker to use a new value if it the second rule's match value finds a hit (and only 10 matches, 0-9, are permitted) for the results (yield) from applying the first rule.

Consider the example of the following regular expression entry made for a **match-value** parameter:'Trunk(.+)', which might be set as that match value in the first rule you configure. Given a SIP element rule called uri-param and the param-name tgid, it can yield two values:

- Grouping 0—The entire matching string (Trunk1)
- Grouping 1—The grouping (1)

In turn, these groupings can be referenced in an element rule by using this syntax:

```
$<header rule name >.$<element rule name.$<value>
```

Additional syntax options that can be used with this feature are:

- $headerName['['index']']
- $headerName['['index']'][.$index]
- $headerName['['index']'][.$elementName]
- $headerName['['index']'][.$elementName][.$index]

Guidelines for Header and Element Rules

Header rules and element rules share these guidelines:

- References to groupings that do not exist result in an empty string.
- References to element rule names alone result in a Boolean condition of whether the expression matched or not.
- A maximum of ten matches are allowed for a regular expression. Match 0 (grouping 0) is always the match of the entire matching string; subsequent numbers are the results for other groups that match.

Splitting and Joining Headers

To simplify header manipulation processes, the Oracle Enterprise Communications Broker provides a means of combining or breaking apart header strings that actually consist of multiple headers. An example application would be to separate headers that another SIP device joined together into a single string. This would allow header manipulation to work on each distinct header, after which the system could re-combine the headers, making the forwarded output consistent with the initial message.

Some SIP devices combine multiple headers into a single header, with each distinct header separated by a comma. This is not precluded by RFC 3261. The user can configure a header manipulation to separate these headers prior to performing the manipulation using the **split-headers** command. The user can also configure the system to join headers together after a manipulation is complete using the **join-headers** command. Split and join functions do not have to co-exist within a single header manipulation.

Precedence

The Oracle Enterprise Communications Broker applies SIP header rules in the order you have entered them. This guards against the Oracle Enterprise Communications Broker removing data that might be used in the other header rules.

This ordering also provides you with ways to use manipulations strategically. For example, you might want to use two rules if you want to store the values of a regular expression. The first rule would store the value of a matched regular expression, and the second could delete the matched value.

In addition to taking note of the order in which header rules are configured, you now must also configure a given header rule prior to referencing it. For example, you must create Rule1 with the action store for the Contact header BEFORE you can create Rule2 which uses the stored value from the Contact header.

Duplicate Header Names

If more than one header exists for a configured header-name, the ECB stores each value in an array whose index starts at 0. To reference those values, use the syntax `$<header-name>[<index>]`.

Add a trailing `[<index>]` value after the header-name parameter to represent the specific instance of the header on which to operate. Additional stored header values are indexed in the order in which they appear within the SIP message, and there is no limit to the index. The ECB takes no action if the header does not exist.

In addition to numerical values, possible index values are:

- ~ The ECB references the first matching header.
- * The ECB references all headers.
- ^ The ECB references the last stored header in the header rule.

Note that the header instance functionality has no impact on HMR's add action, and you cannot use this feature to insert headers into a specific location. Headers are added to the end of the list, except that Via headers are added to the top.

Performing HMR on a Specific Header

HMR has been enhanced so that you can now operate on a specific instance of a given header. The syntax you use to accomplish this is similar to that you used to refer to a specific header rule stored value instance.

Using the header-name parameter, you can now add a trailing [<index>] value after the header name. This [<index>] is a numerical value representing the specific instance of the header on which to operate. However, the Oracle Enterprise Communications Broker takes no action if the header does not exist. You can also use the caret (^) to reference the last header of that type (if there are multiple instances)

The count for referencing is zero-based, meaning that the first instance of the header counts as 0.

Note that the header instance functionality has no impact on HMR's add action, and you cannot use this feature to insert headers into a specific location. Headers are added to the end of the list, except that Via headers are added to the top.

Multiple SIP HMR Sets

In general you use SIP HMR by configuring rules and then applying those rules to session agents, realms, or SIP interfaces in the inbound or outbound direction. In addition, the Oracle Enterprise Communications Broker has a set method for how certain manipulation rules take precedence over others. For instance, inbound SIP manipulation rules defined in a session agent take precedence over any configured for a realm, and the rules for a realm take precedence over SIP interface manipulation rules.

The multiple SIP HMR feature gives you the ability to:

- Apply multiple inbound and outbound manipulations rules to a SIP message
- Provision the order in which the Oracle Enterprise Communications Broker applies manipulation rules

The **action** parameter in the header rules configuration now takes the value sip-manip. When you set the parameter to sip-manip, you then configure the **new-value** parameter with the name of a SIP manipulation rule that you want to invoke. The values for the **match-value**, **comparison-type**, and **methods** parameters for invoked rule are all supported. This means that the manipulation defined by the rules identified in the **new-value** parameter are carried out when the values for the **match-value**, **comparison-type**, and **methods** parameters are true.

The relationship between manipulation rules and manipulation rule sets is created once you load your configuration, meaning that the order in which you enter them does not matter. It also means that the Oracle Enterprise Communications Broker cannot dynamically perform validation as you enter rules, so you should use the ACLI **verify-config** command to confirm your manipulation rules contain neither invalid nor circular references. Invalid references are those that point to SIP manipulation rules that do not exist, and circular references are those that create endless loops of manipulation rules being carried out over and over. If you load a configuration exhibiting either of these issues, the Oracle Enterprise Communications Broker forces the action value for the rule to **none** and the rule will not be used.

MIME Support

Using the SIP HMR feature set, you can manipulate MIME types in SIP message bodies. While you can manipulate the body of SIP messages or a specific content type using other iterations of SIP HMR, this version gives you the power to change the MIME attachment of a specific type within the body by using regular expressions. To achieve this, you use the **find-replace-all** action type, which enables the search for a particular string and the replacement of all matches for that type. Although you use **find-replace-all** to manipulate MIME attachments, it can also be used to achieve other goals in SIP HMR.

Note that using **find-replace-all** might consume more system resources than other HMR types. Therefore this powerful action type should only be used when another type cannot perform the type of manipulation you require.

Manipulating MIME Attachments

Set the action type to find-replace-all to modify MIME attachments.

To manipulate a particular portion of the MIME attachment, for example when removing a certain attribute within the Content-Type of `application/sdp`, the ECB needs to search the content multiple times because:

- SDP can have more than one media line

- The SIP message body can contain more than one `application/sdp`.

When the action type is find-replace-all, the ECB treats the match-value as a regular expression and binds the comparison-type to pattern-rule, even if comparison-type is set to some other value. This type of action is both a comparison and action: for each regular expression match within the supplied string, the ECB substitutes the new value for that match.

Use subgroups to replace portions of the regular expression rather than the entire matched expression. The subgroup replacement syntax is formed by adding the string `[[:n:]]` to the end of the regular expression—where `n` is a number between 0 and 9. For example, setting the following parameters

```
action                  find-replace-all
match-value             sip:(user)@host[[:1:]]
new-value               bob
```

creates a new rule to replace only the user portion of the URI that searches for the regular expression and replaces all instances of the user subgroup with the value bob.

Setting the following parameters

```
action                  find-replace-all
match-value             0
new-value               1
```

creates a new rule to recursively replace all the 0 digits in a telephone number with 1. With this rule the user portion of a URI—or for any other string—with a value 1-781-308-4400 would be replaced as 1-781-318-4411.

If you leave the **new-value** parameter blank for **find-replace-all**, the ECB replaces the matched sub-group with an empty string—an equivalent of deleting the sub-group match. You can also replace empty sub-groups, which is like inserting a value within the second sub-group match. For example, `user()@host.com[[:1:]]` with a configured new-value `_bob` yields `user_bob@host.com`.

Setting **find-replace-all** disables the following **parameter-type** values: **uri-param-name**, **uri-header-name**, and **header-param-name**. These values are unusable because the ECB only uses case-sensitive matches for the match-value to find the parameter name within the URI. Since it can only be found by exact match, the ECB does not support finding and replacing that parameter.

Escaped Characters

SIP HMR's support for escaped characters allows for searches for values you would be unable to enter yourself. Because they are necessary to MIME manipulation, support for escaped characters now includes:

- \f
- \n
- \r
- \t
- \v

New Reserved Word

To allow you to search for carriage returns an new lines, the SIP HMR MIME feature also adds support for the reserved word $CRLF. Because you can search for these value and replace them, you also must be able to add them back in when necessary. Configuring $CRLF in the **new-value** parameter always resolves to /r/n, which you normally cannot otherwise enter through the ACLI.

About the MIME Value Type

Introduced to modify the MIME attachment, SIP HMR supports a **mime** value for the **type** parameter in the element rules configuration. Like the **status-code** and **reason-phrase** values, you can only use the **mime** type value against a specific header—which in this case, is Content-Type.

When you set the element rule type to **mime**, you must also configure the **parameter-name** with a value. This step is a requirement because it sets the content-type the Oracle Enterprise Communications Broker manipulates in a specific part of the MIME attachment. You cannot leave this parameter blank; the Oracle Enterprise Communications Broker does not let you save the configuration if you do. When you use the **store** action on a multi-part MIME attachment that has different attachment types, the Oracle Enterprise Communications Broker stores the final instance of the content-type because it does not support storing multiple instances of element rule stored values.

In the event you do not know the specific content-type where the Oracle Enterprise Communications Broker will find the **match-value**, you can wildcard the **parameter-name** by setting with the asterisk (*) as a value. You cannot, however, set partial content-types (i.e., application/*). So configured, the Oracle Enterprise Communications Broker loops through the MIME attachment's content types.

You can set the additional **action** types listed in this table with the described result:

Action Type	Description
delete-element	Removes the matched mime-type from the body. If this is the last mime-type within in message body, the Oracle Enterprise Communications Broker removes the Content-Type header.
delete-header	Removes all body content and removes the Content-Type header.
replace	Performs a complete replacement of the matched mime-type with the new-value you configure.
find-replace-all	Searches the specifies mime-type's contents and replaces all matching regular expressions with the new-value you configure
store	Stores the final instance of the content-type (if there are multi-part MIME attachments of various attachment types)
add	Not supported

MIME manipulation does not support manipulating headers in the individual MIME attachments. For example, the Oracle Enterprise Communications Broker cannot modify the Content-Type given a portion of a message body like this one:

```
--boundary-1
Content-Type: application/sdp
v=0
o=use1 53655765 2353687637 IN IP4 192.168.1.60
s=-
```

```
c=IN IP4 192.168.1.60
t=0 0
m=audio 10000 RTP/AVP 8
a=rtpmap:8 PCMA/8000/1
a=sendrecv
a=ptime:20
a=maxptime:200
```

Back Reference Syntax

You can use back reference syntax in the **new-value** parameter for header and element rules configurations. Denoted by the use of $1, $2, $3, etc. (where the number refers to the regular expression's stored value), you can reference the header and header rule's stored value without having to use the header rule's name. It instead refers to the stored value of this rule.

For example, when these settings are in place:

- header-rule=changeHeader
- action=manipulate
- match-value=(.+)([^;])

you can set the **new-value** as sip:$2 instead of **sip:$changeHeader.$2**.

You can use the back reference syntax for:

- Header rule **action**s **manipulate** and **find-replace-all**
- Element rule **action**s **replace** and **find-replace-all**

Using back reference syntax simplifies your configuration steps because you do not need to create a store rule and then manipulate rule; the manipulate rule itself performs the store action if the **comparison-type** is set to **pattern-rule**.

Notes on the Regular Expression Library

In the regular expression library, the dot (.) character no longer matches new lines or carriage returns. Conversely, the not-dot does match new lines and carriage returns. This change provides a safety mechanism preventing egregious backtracking of the entire SIP message body when there are no matches. Thus, the Oracle Enterprise Communications Broker reduces backtracking to a single line within the body. In addition, there is now support for:

Syntax	Description
\s	Whitespace
\S	Non-whitespace
\d	Digits
\D	Non-digits
\R	Any \r, \n, \r\n
\w	Word
\W	Non-word
\A	Beginning of buffer
\Z	End of buffer
\	Any character including newline, in the event that the dot (.) is not

In addition, there is:

- Escaped character shortcuts (\w\W\S\s\d\D\R) operating inside brackets [...]

SIP Message-Body Separator Normalization

The Oracle Enterprise Communications Broker supports SIP with Multipurpose Internet Mail Extension (MIME) attachments — up to a maximum payload size of 64KB — and has the ability to allow more than the required two CRLFs between the SIP message headers and the multipart body's first boundary. The first two CRLFs that appear in all SIP messages signify the end of the SIP header and the separation of the header and body of the message, respectively. Sometimes additional extraneous CRLFs can appear within the preamble before any text.

The Oracle Enterprise Communications Broker works by forwarding received SIP messages regardless of whether they contain two or more CRLFs. Although three or more CRLFs are legal, some SIP devices do not accept more than two.

The solution to ensuring all SIP devices accept messages sent from the Oracle Enterprise Communications Broker is to strip all CRLFs located at the beginning of the preamble before the appearance of any text, ensuring that there are no more than two CRLFs between the end of the last header and the beginning of the body within a SIP message. You enable this feature by adding the new stripPreambleCrlf option to the global SIP configuration.

To enable the stripping of CRLFs in the preamble, add the **+stripPreambleCrlf** option to SIP Options.

SIP Header Pre-Processing HMR

By default, the Oracle Enterprise Communications Broker performs in-bound SIP manipulations after it carries out header validation. Adding the **inmanip-before-validate** option in the global SIP configuration allows the Oracle Enterprise Communications Broker to perform HMR on received requests prior to header validation. Because there are occaisional issues with other SIP implementations—causing invalid headers to be used in messages they send to the Oracle Enterprise Communications Broker—it can be beneficial to use HMR to remove or repair these faulty headers before the request bearing them are rejected.

When configured to do so, the Oracle Enterprise Communications Broker performs pre-validation header manipulation immediately after it executes the top via check. Inbound SIP manipulations are performed in order of increasing precedence: SIP interface, realm, and session agent.

The fact that the top via check happens right before the Oracle Enterprise Communications Broker carries out pre-validation header manipulations means that you cannot use this capability to repairs the first via header if it is indeed invalid. If pre-validation header manipulation were to take place at another time during processing, it would not be possible to use it for SIP session agents. The system learns of matching session agents after top via checking completes.

For logistical reasons, this capability does not extend to SIP responses. Inbound manipulation for responses cannot be performed any sooner that it does by default, a time already preceding any header validation.

To enable SIP header pre-processing, add the **+inmanip-before-validate** option to SIP Options.

Best Practices

This section lists practices that Oracle recommends you follow for successful implementation of this feature.

- Define all storage rules first.
  This recommendation is made because each subsequent header rule processes against the same SIP message, so each additional header rules works off of the results from the application of the rule that precedes it.

  In general, you want to store values from the original SIP header rather than from the iteratively changed versions.
- Implement rules at the element rule rather than the header rule level.
  Header rules should only be a container for element rules.
- When you are creating rules to edit a header, add additional element rules to modify a single header rather than try to create multiple header rules each with one element rule. That is, create multiple element rules within a header rule rather than creating multiple header rules.
- Do not use header or element rule names that are all capital letters (i.e., $IP_ADDRESS). Capitals currently refer to predefined rules that are used as macros, and they might conflict with a name that uses capital letters.

About Regular Expressions

Two of the most fundamental ideas you need to know in order to work with regular expressions and with this feature are:

- Regular expressions are a way of creating strings to match other string values.
- You can use groupings in order to create stored values on which you can then operate.

To learn more about regex, you can visit the following Web site, which has information and tutorials that can help to get you started:http://www.regular-expressions.info/.

Many of the characters you can type on your keyboard are literal, ordinary characters—they present their actual value in the pattern. Some characters have special meaning, however, and they instruct the regex function (or engine which interprets the expressions) to treat the characters in designated ways. The following table outlines these “special characters” or metacharacters.

Character	Name	Description
.	dot	Matches any one character, including a space; it will match one character, but there must be one character to match. Literally a . (dot) when bracketed ([]), or placed next to a \ (backslash).
*	star/asterisk	Matches one or more preceding character (0, 1, or any number), bracketed carrier class, or group in parentheses. Used for quantification. Typically used with a . (dot) in the format .* to indicate that a match for any character, 0 or more times. Literally an * (asterisk) when bracketed ([]).
+	plus	Matches one or more of the preceding character, bracketed carrier class, or group in parentheses. Used for quantification. Literally a + (plus sign) when bracketed ([]).

Character	Name	Description
|	bar/vertical bar/pipe	Matches anything to the left or to the right; the bar separates the alternatives. Both sides are not always tried; if the left does not match, only then is the right attempted. Used for alternation.
{	left brace	Begins an interval range, ended with } (right brace) to match; identifies how many times the previous singles character or group in parentheses must repeat. Interval ranges are entered as minimum and maximums ({minimum,maximum}) where the character/group must appear a minimum of times up to the maximum. You can also use these character to set magnitude, or exactly the number of times a character must appear; you can set this, for example, as the minimum value without the maximum ({minimum,}).
?	question mark	Signifies that the preceding character or group in parentheses is optional; the character or group can appear not at all or one time.
^	caret	Acts as an anchor to represent the beginning of a string.
$	dollar sign	Acts as an anchor to represent the end of a string.
[	left bracket	Acts as the start of a bracketed character class, ended with the] (right bracket). A character class is a list of character options; one and only on of the characters in the bracketed class must appear for a match. A - (dash) in between two character enclosed by brackets designates a range; for example [a-z] is the character range of the lower case twenty-six letters of the alphabet. Note that the] (right bracket) ends a bracketed character class unless it sits directly next to the [(left bracket) or the ^ (caret); in those two cases, it is the literal character.
(	left parenthesis	Creates a grouping when used with the) (right parenthesis). Groupings have two functions: They separate pattern strings so that a whole string can have special characters within it as if it were a single character. They allow the designated pattern to be stored and referenced later (so that other operations can be performed on it).

Expression Building Using Parentheses

You can now use parentheses (())when you use HMR to support order of operations and to simplify header manipulation rules that might otherwise prove complex. This means that expressions such as (sip + urp) - (u + rp) can now be evaluated to sip. Previously, the same expression would have evaluated to sipurprp. In addition, you previously would have been required to create several different manipulation rules to perform the same expression.

Configuration Examples

This section shows you several configuration examples for HMR. This section shows the configuration for the various rules that the Oracle Enterprise Communications Brokerapplied, and sample results of the manipulation. These examples present configurations as an entire list of fields and settings for each ruleset, nested header rules and nested element rules. If a field does not have any operation within the set, the field is shown with the setting at the default or blank.

Example 1 Removing Headers

For this manipulation rule, the Oracle Enterprise Communications Broker removes the Custom header if it matches the pattern rule. It stores the defined pattern rule for the goodBye header. Finally, it removes the goodBye header if the pattern rule from above is a match.

This is a sample of the configuration:

```
sip-manipulation
        name                            removeHeader
        header-rule
                name                            removeCustom
                header-name                     Custom
                action                          delete
                comparison-type                 boolean
                match-value                     ^This is my.*
                msg-type                        request
                new-value
                methods                         INVITE
        header-rule
                name                            goodByeHeader
                header-name                     Goodbye
                action                          store
comparison-type     boolean
                match-value                     ^Remove (.+)
                msg-type                        request
                new-value
                methods                         INVITE
header-rule
                name                            goodBye
action                          delete
                comparison-type                 pattern-rule
                match-value                     $goodByeHeader
                msg-type                        request
                new-value
                methods                         INVITE
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK0g639r10fgc0aakk26s1.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDc1rm601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SDc1rm601-d01673bcacfcc112c053d95971330335-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
```

```
Params: sipp <sip:sipp2@192.168.1.60:5060>
Edit: disp <sip:user@192.168.1.60:5060>
Max-Forwards: 69
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 140
```

Example 2 Manipulating the Request URI

For this manipulation rules, the Oracle Enterprise Communications Broker stores the URI parameter tgid in the Request URI. Then if the pattern rule matches, it adds a new header (x-customer-profile) with the a new header value tgid to the URI parameter in the request URI.

This is a sample of the configuration:

```
sip-manipulation
        name                            CustomerTgid
        header-rule
                name                            ruriRegex
                header-name                     request-uri
                action                          store
                comparison-type                 pattern-rule
                match-value
                msg-type                        request
new-value
                methods                         INVITE
                element-rule
                        name                            tgidParam
                        parameter-name                  tgid
                        type                            uri-param
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
header-rule
                name                            addCustomer
                header-name                     X-Customer-Profile
                action                          add
                comparison-type                 pattern-rule
                match-value                     $ruriRegex.$tgidParam
                msg-type                        request
                new-value                       $ruriRegex.$tgidParam.$0
                methods                         INVITE
header-rule
                name                            delTgid
                header-name                     request-uri
                action                          manipulate
                comparison-type                 pattern-rule
                match-value                     $ruriRegex.$tgidParam
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            tgidParam
                        parameter-name                  tgid
                        type                            uri-param
                        action                          delete-element
                        match-val-type                  any
                        comparison-type                 case-sensitive
```

```
                        match-value                     $ruriRegex.$tgidParam.$0
                        new-value
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060 SIP/2.0
    Message Header
Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK0g6plv3088h03acgh6c1.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDc1rg601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SDc1rg601-f125d8b0ec7985c378b04cab9f91cc09-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my custom header
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060>
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
        Content-Length: 140
        X-Customer-Profile: 123
```

Example 3 Manipulating a Header

For this manipulation rule, the Oracle Enterprise Communications Brokerstores the pattern matches for the Custom header, and replaces the value of the Custom header with a combination of the stored matches and new content.

This is a sample of the configuration:

```
sip-manipulation
        name                            modCustomHdr
        header-rule
                name                            customSearch
                header-name                     Custom
                action                          store
                comparison-type                 pattern-rule
                match-value                     (This is my )(.+)( header)
                msg-type                        request
                new-value
                methods                         INVITE
header-rule
                name                            customMod
                header-name                     Custom
                action                          manipulate
                comparison-type                 pattern-rule
                match-value                     $customSearch
                msg-type                        request
                new-value
methods                         INVITE
                element-rule
                        name                            hdrVal
                        parameter-name                  hdrVal
                        type                            header-value
                        action                          replace
                        match-val-type                  any
                        comparison-type                 case-sensitive
```

```
                        match-value
new-value                       $customSearch.$1+edited+$customSearch.$3
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK20q2s820boghbacgs6o0.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDe1ra601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SDe1ra601-4bb668e7ec9eeb92c783c78fd5b26586-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my edited header
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060>
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
        Content-Length: 140
```

Example 4 Storing and Using URI Parameters

For this manipulation rule, the Oracle Enterprise Communications Broker stores the value of the URI parameter tag from the From header. It also creates a new header FromTag with the header value from the stored information resulting from the first rule.

This is a sample of the configuration:

```
sip-manipulation
        name                            storeElemParam
        header-rule
                name                            Frohmr
                header-name                     From
                action                          store
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            elementRule
                        parameter-name                  tag
                        type                            uri-param
                        action                          store
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value
header-rule
                name                            newHeader
                header-name                     FromTag
                action                          add
                comparison-type                 pattern-rule
                match-value                     $FromHR.$elementRule
                msg-type                        any
                new-value                       $FromHR.$elementRule.$0
                methods
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK4oda2e2050ih7acgh6c1.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDf1re601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SDf1re601-f85059e74e1b443499587dd2dee504c2-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my custom header
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060>
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
Content-Length: 140
        FromTag: 1
```

Example 5 Manipulating Display Names

For this manipulation rule, the Oracle Enterprise Communications Broker sores the display name from the Display header. It replaces the two middle characters of the original display name with a new string. Then is also replaces the From header's display name with "abc 123" if it matches sipp.

This is a sample of the configuration:

```
sip-manipulation
        name                            modDisplayParam
        header-rule
                name                            storeDisplay
                header-name                     Display
                action                          store
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            displayName
                        parameter-name                  display
                        type                            uri-display
                        action                          store
                        match-val-type                  any
comparison-type                 pattern-rule
                        match-value                     (s)(ip)(p )
                        new-value
header-rule
                name                            modDisplay
                header-name                     Display
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
```

```
                element-rule
                        name                            modRule
                        parameter-name                  display
                        type                            uri-display
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $storeDisplay.$displayName
                        new-value                       $storeDisplay.
$displayName.$1+lur+$storeDisplay.$displayName.$3
header-rule
                name                            modFrom
                header-name                     From
                action                          manipulate
                comparison-type                 pattern-rule
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            fromDisplay
                        parameter-name
                        type                            uri-display
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     sipp
                        new-value                       "\"abc 123\" "
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK681kot109gp04acgs6o0.1
        From: "abc 123" <sip:sipp@192.168.1.60:5060>;tag=SD79ra601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SD79ra601-a487f1259e2370d3dbb558c742d3f8c4-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my custom header
        Display: slurp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060>
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
        Content-Length: 140
```

Example 6 Manipulating Element Parameters

For this more complex manipulation rule, the Oracle Enterprise Communications Broker:

- From the Display header, stores the display name, user name, URI parameter up, and header parameter hp
- Adds the header parameter display to the Params header, with the stored value of the display name from the first step

- Add the URI parameter user to the Params header, with the stored value of the display name from the first step
- If the URI parameter match succeeds in the first step, replaces the URI parameter up with the Display header with the value def
- If the header parameter match succeeds in the first step, deletes the header parameter hp from the Display header

This is a sample of the configuration:

```
sip-manipulation
        name                            elemParams
        header-rule
                name                            StoreDisplay
                header-name                     Display
                action                          store
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            displayName
                        parameter-name
                        type                            uri-display
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
element-rule
                        name                            userName
                        parameter-name                  user
                        type                            uri-user
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
element-rule
                        name                            uriParam
                        parameter-name                  up
                        type                            uri-param
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
element-rule
                        name                            headerParam
                        parameter-name                  hp
                        type                            header-param
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
        header-rule
                name                            EditParams
                header-name                     Params
                action                          manipulate
```

```
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            addHeaderParam
                        parameter-name                  display
                        type                            header-param
                        action                          add
match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value                       $StoreDisplay.
$displayName.$0
                element-rule
                        name                            addUriParam
                        parameter-name                  user
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
              new-value
$StoreDisplay.$userName.$0
        header-rule
                name                            EditDisplay
                header-name                     Display
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            replaceUriParam
                        parameter-name                  up
                        type                            uri-param
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $StoreDisplay.$uriParam
                        new-value                       def
                element-rule
                        name                            delHeaderParam
                        parameter-name                  hp
                        type                            header-param
                        action                          delete-element
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value $StoreDisplay.$headerParam
                        new-value
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK7okvei0028jgdacgh6c1.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SD89rm601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SD89rm601-b5b746cef19d0154cb1f342cb04ec3cb-06a3gu0
```

```
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
Goodbye: Remove Me
Custom: This is my custom header
Display: sipp <sip:user@192.168.1.60:5060;up=def>
Params: sipp <sip:sipp1@192.168.1.60:5060;user=user>;display=sipp
Params: sipp <sip:sipp2@192.168.1.60:5060;user=user>;display=sipp
Edit: disp <sip:user@192.168.1.60:5060>
Max-Forwards: 69
Subject: Performance Test
Content-Type: application/sdp
Content-Length: 140
```

Example 7 Accessing Data from Multiple Headers of the Same Type

For this manipulation rule, the Oracle Enterprise Communications Broker stores the user name from the Params header. It then adds the URI parameter c1 with the value stored from the first Params header. Finally, it adds the URI parameter c2 with the value stored from the second Params header.

This is a sample of the configuration:

```
sip-manipulation
        name                            Params
        header-rule
                name                            storeParams
                header-name                     Params
                action                          store
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            storeUserName
                        parameter-name                  user
                        type                            uri-user
                        action                          store
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value
header-rule
                name                            modEdit
                header-name                     Edit
                action                          manipulate
                comparison-type                 pattern-rule
                match-value
                msg-type                        request
                new-value
methods                         INVITE
                element-rule
                        name                            addParam1
                        parameter-name                  c1
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
```

```
                        match-value
                        new-value                       $storeParams[0].
$storeUserName.$0
                element-rule
                        name                            addParam2
                        parameter-name                  c2
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value                       $storeParams[1].
$storeUserName.$0
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK9g855p30cos08acgs6o0.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SD99ri601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SD99ri601-6f5691f6461356f607b0737e4039caec-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my custom header
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060;c1=sipp1;c2=sipp2>
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
        Content-Length: 140
```

Example 8 Using Header Rule Special Characters

For this manipulation rule, the Oracle Enterprise Communications Broker:

- Stores the header value of the Params header with the given pattern rule, and stores both the user name of the Params header and the URI parameter abc
- Adds the URI parameter lpu with the value stored from the previous Params header
- If any of the Params headers match the pattern rule defined in the first step, adds the URI parameter apu with the value aup
- If all of the Params headers match the pattern rule defined in the first step, adds the URI parameter apu with the value apu
- If the first Params headers does not match the pattern rule for storing the URI parameter defined in the first step, adds the URI parameter not with the value 123
- If the first Params headers matches the pattern rule for storing the URI parameter defined in the first step, adds the URI parameter yes with the value 456

This is a sample of the configuration:

```
sip-manipulation
        name                            specialChar
        header-rule
                name                            searchParams
```

```
                header-name                     Params
                action                          store
                comparison-type                 pattern-rule
                match-value                     .*sip:(.+)@.*
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            userName
                        parameter-name
                        type                            uri-user
                        action                          store
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value
element-rule
                        name                            emptyUriParam
                        parameter-name                  abc
                        type                            uri-param
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value
                        new-value
header-rule
                name                            addUserLast
                header-name                     Edit
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            lastParamUser
                        parameter-name                  lpu
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value $searchParams[^].$userName.$0
                element-rule
                        name                            anyParamUser
                        parameter-name                  apu
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $searchParams[~]
                        new-value                       aup
                element-rule
                        name                            allParamUser
                        parameter-name                  apu
                        type                            header-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $searchParams[*]
                        new-value                       apu
```

```
                element-rule
                        name                            notParamYes
                        parameter-name                  not
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     !$searchParams.
$emptyUriParam
                        new-value                       123
                element-rule
                        name                            notParamNo
                        parameter-name                  yes
                        type                            uri-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $searchParams.
$emptyUriParam
                        new-value                       456
```

This is a sample of the result:

```
Request-Line: INVITE sip:service@192.168.200.60:5060;tgid=123 SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.200.61:5060;branch=z9hG4bK681m9t30e0qh6akgj2s1.1
        From: sipp <sip:sipp@192.168.1.60:5060>;tag=SDchrc601-1
        To: sut <sip:service@192.168.1.61:5060>
        Call-ID: SDchrc601-fcf5660a56e2131fd27f12fcbd169fe8-06a3gu0
        CSeq: 1 INVITE
        Contact: <sip:sipp@192.168.200.61:5060;transport=udp>
        Goodbye: Remove Me
        Custom: This is my custom header
        Display: sipp <sip:user@192.168.1.60:5060;up=abc>;hp=123
        Params: sipp <sip:sipp1@192.168.1.60:5060>
        Params: sipp <sip:sipp2@192.168.1.60:5060>
        Edit: disp <sip:user@192.168.1.60:5060;lpu=sipp2;apu=aup;not=123>;apu=apu
        Max-Forwards: 69
        Subject: Performance Test
        Content-Type: application/sdp
        Content-Length: 140
```

Example 9 Status-Line Manipulation

This section shows an HMR configuration set up for status-line manipulation.

Given that the object of this example is to drop the 183 Session Progress response when it does not have SDP, your SIP manipulation configuration needs to:

1. Search for the 183 Session Progress response
2. Determine if the identified 183 Session Progress responses contain SDP; the Oracle Enterprise Communications Broker searches the 183 Session Progress responses where the content length is zero
3. If the 183 Session Progress response does not contain SDP, change its status code to 699
4. Drop all 699 responses

```
sip-manipulation
        name                            manip
        description
```

```
        header-rule
                name                            IsContentLength0
                header-name                     Content-Length
                action                          store
                comparison-type                 pattern-rule
                match-value                     0
                msg-type                        reply
                new-value
                methods
        header-rule
                name                            is183
                header-name                     @status-line
                action                          store
                comparison-type                 pattern-rule
                match-value
                msg-type                        reply
                new-value
                methods
                element-rule
name                            is183Code
                        parameter-name
                        type                            status-code
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     183
                        new-value
        header-rule
                name                            change183
                header-name                     @status-line
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        reply
                new-value
                methods
                element-rule
                        name                            make199
                        parameter-name
                        type                            status-code
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $IsContentLength0 &
$is183.$is183Code
                        new-value                       199

sip-interface     options dropResponse=699
```

Example 10 Use of SIP HMR Sets

The following example shows the configuration for SIP HMR with one SIP manipulation configuration loading another SIP manipulation configuration. The goals of this configuration are to:

- Add a new header to an INVITE
- Store the user portion of the Request URI
- Remove all Route headers from the message only if the Request URI is from a specific user

```
sip-manipulation
        name                            deleteRoute
        description                     delete all Route Headers
        header-rule
                name                            deleteRoute
                header-name                     Route
                action                          delete
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
sip-manipulation
        name                            addAndDelete
        description                     Add a New header and delete Route
headers
        header-rule
                name                            addHeader
                header-name                     New
                action                          add
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value                       "Some Value"
                methods                         INVITE
        header-rule
                name                            storeRURI
                header-name                     request-uri
                action                          store
                comparison-type                 pattern-rule
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            storeUser
                        parameter-name
                        type                            uri-user
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     305.*
                        new-value
        header-rule
                name                            deleteHeader
                header-name                     request-uri
                action                          sip-manip
                comparison-type                 Boolean
                match-value                     $storeRURI.$storeUser
                msg-type                        request
                new-value                       deleteRoute
                methods                         INVITE
```

Example 11 Use of Remote and Local Port Information

The following example shows the configuration for remote and local port information. The goals of this configuration are to:

- Add LOCAL_PORT as a header parameter to the From header
- Add REMOTE_PORT as a header parameter to the From header

```
sip-manipulation
        name                            addOrigIp
        description
        header-rule
                name                            addIpParam
                header-name                     From
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            addIpParam
                        parameter-name                  newParam
                        type                            header-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value                       $LOCAL_IP
                element-rule
                        name                            addLocalPort
                        parameter-name                  lport
                        type                            header-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value                       $LOCAL_PORT
                element-rule
                        name                            addRemotePort
                        parameter-name                  rport
                        type                            header-param
                        action                          add
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value
                        new-value                       $REMOTE_PORT
```

Example 12 Response Status Processing

Given that the object of this example is to drop the 183 Session Progress response when it does not have SDP, your SIP manipulation configuration needs to:

1. Search for the 183 Session Progress response
2. Determine if the identified 183 Session Progress responses contain SDP; the Oracle Enterprise Communications Broker searches the 183 Session Progress responses where the content length is zero
3. If the 183 Session Progress response does not contain SDP, change its status code to 699
4. Drop all 699 responses

```
sip-manipulation
        name                            manip
        description
        header-rule
                name                            IsContentLength0
                header-name                     Content-Length
```

```
                action                          store
                comparison-type                 pattern-rule
                match-value                     0
                msg-type                        reply
                new-value
                methods
        header-rule
                name                            is183
                header-name                     @status-line
                action                          store
                comparison-type                 pattern-rule
                match-value
                msg-type                        reply
                new-value
                methods
                element-rule
                        name                            is183Code
                        parameter-name
                        type                            status-code
                        action                          store
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     183
                        new-value
        header-rule
                name                            change183
                header-name                     @status-line
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        reply
                new-value
                methods
                element-rule
                        name                            make699
                        parameter-name
                        type                            status-code
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     $IsContentLength0 &
$is183.$is183Code
                        new-value                       699
sip-interface
    options dropResponse=699
```

The following four configuration examples are based on the this sample SIP INVITE:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length:  466
--boundary
Content-Type: application/sdp
v=0
```

```
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 18
a=rtpmap:8 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=ptime:20
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
a=ptime:30
--boundary--
```

Example 13 Remove a Line from SDP

In this example, the SIP manipulation is configured to remove all p-time attributes from the SDP.

```
sip-manipulation
        name                            removePtimeFromBody
        description                   removes ptime attribute from all bodies
        header-rule
                name                            CTypeManp
                header-name                     Content-Type
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            remPtime
                        parameter-name                  application/sdp
                        type                            mime
                        action                          find-replace-all
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value                     a=ptime:[0-9]{1,2}(\n|\r
\n)
                        new-value
```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
```

```
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length:  466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 18
a=rtpmap:18 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
--boundary-
```

Example 14 Back Reference Syntax

In this sample of back-reference syntax use, the goal is to change the To user. The SIP manipulation would be configured like the following:

```
sip-manipulation
        name                            changeToUser
        description                   change user in the To header
        header-rule
                name                            ChangeHeader
                header-name                     To
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        request
                new-value
                methods                         INVITE
                element-rule
                        name                            replaceValue
                        parameter-name
                        type                            header-value
                        action                          replace
                        match-val-type                  any
                        comparison-type                 pattern-rule
                        match-value                     (.+)(service)(.+)
                        new-value                       $1+Bob+$3
```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:Bob@192.168.1.61:5060>
```

```
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length:  466
…
…
…
```

Example 15 Change and Remove Lines from SDP

In this sample of changing and removing lines from the SDP, the goal is to convert the G.729 codec to G.729a. The SIP manipulation would be configured like the following:

```
sip-manipulation
        name                            std2prop-codec-name
        description                     rule to translate standard to proprietary
codec name
        header-rule
                name                            CTypeManp
                header-name                     Content-Type
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        any
                new-value
                methods
                element-rule
                        name                            g729-annexb-no-std2prop
                        parameter-name                  application/sdp
                        type                            mime
                        action                          find-replace-all
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value                     a=rtpmap:[0-9]{1,3}
(G729/8000/1\r\na=fmtp:[0-9]{1,3} annexb=no)[[:1:]]
                        new-value                       G729a/8000/1
```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length:  466
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 8
```

```
a=rtpmap:18 G729a/8000/1
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263a/90000
--boundary-
```

Example 16 Change and Add New Lines to the SDP

In this sample of changing and adding lines from the SDP, the goal is to convert non-standard codec H.263a to H.263. The SIP manipulation would be configured like the following:

```
sip-manipulation
        name                            prop2std-codec-name
        description                     rule to translate proprietary to standard
codec name
        header-rule
                name                            CodecManp
                header-name                     Content-Type
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                msg-type                        any
                new-value
                methods
                element-rule
                        name                            H263a-prop2std
                        parameter-name                  application/sdp
                        type                            mime
                        action                          find-replace-all
                        match-val-type                  any
                        comparison-type                 case-sensitive
                        match-value                     a=rtpmap:([0-9]{1,3})
H263a/.*\r\n
                        new-value                       a=rtpmap:+$1+"
H263/90000"+$CRLF+a=fmtp:+$1+" QCIF=4"+$CRLF
```

The result of manipulating the original SIP INVITE (shown above) with the configured SIP manipulation is:

```
INVITE sip:service@192.168.1.61:5060 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.60:5060;branch=z9hG4bK-1-0
From: sipp <sip:sipp@192.168.1.60:5060>;tag=1
To: sut <sip:service@192.168.1.61:5060>
Call-ID: 1-15554@192.168.1.60
CSeq: 1 INVITE
Contact: <sip:sipp@192.168.1.60:5060;user=phone>
Max-Forwards: 70
Content-Type: multipart/mixed;boundary=boundary
Content-Length:  466
--boundary
Content-Type: application/sdp
v=0
```

```
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=audio 12345 RTP/AVP 8
a=rtpmap:18 G729/8000/1
a=fmtp:18 annexb=no
a=sendrecv
a=maxptime:200
--boundary
Content-Type: application/sdp
v=0
o=user1 53655765 2353687637 IN IP4 192.168.1.60
s=-
c=IN IP4 192.168.1.60
t=0 0
m=video 12345 RTP/AVP 34
a=rtpmap:34 H263/90000
a=fmtp:34 QCIF=4
--boundary-
```

Dialog-Matching Header Manipulation

The most common headers to manipulate using HMR are the To-URI and From-URI. Along with the to-tag, from-tag, and Call-ID values, these are also all headers that represent dialog-specific information that must match the UAC and UAS to be considered part of the same dialog. If these parameters are modified through HMR, the results can be that the UAC or UAS rejects messages.

While it is possible to ensure that dialog parameters match correctly using regular HMR, this feature offers a simpler and less error-prone method of doing so.

In addition, this section describes the addition of built-in SIP manipulations defined by Oracle best practices, and a new method of testing your SIP manipulations.

About Dialog-Matching Header Manipulations

The goal of this feature is to maintain proper dialog-matching through manipulation of dialog-specific information using HMR. Two fundamental challenges arise when looking at the issue of correctly parameters manipulating dialog-matching:

- Inbound HMR
- Outbound HMR

The new setting **out-of-dialog** (for the **msg-type** parameter) addresses these challenges by offering an intelligent more of dialog matching of messages for inbound and outbound HMR requests. This is a msg-type parameter, meaning that it becomes matching criteria for operations performed against a message. If you also specify methods (such as REGISTER) as matching criteria, then the rule is further limited to the designated method.

For both inbound and outbound manipulations, using the **out-of-dialog** setting means the message must be a request without a to-tag in order to perform the manipulation.

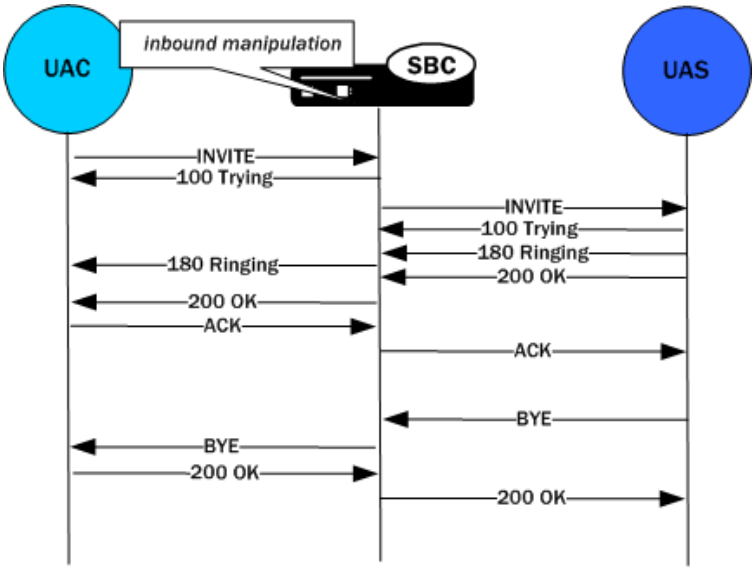
Inbound HMR Challenge

Since inbound manipulations take place before the message reaches the core of Oracle Enterprise Communications Broker SIP processing, the SIP proxy takes the manipulated header

as what was directly received from the client. This can cause problems for requests leaving the Oracle Enterprise Communications Broker for the UAC because the dialog will not match the initial request sent.

So the unmodified header must be cached because for any subsequent request (as in the case of a BYE originating from the terminator; see the diagram below) the Oracle Enterprise Communications Broker might need to restore the original value—enabling the UAC to identify the message correctly as being part of the same dialog. For out-of-dialog requests (when the To, From, or Call-ID headers are modified) the original header will be stored in the dialog when the **msg-type out-of-dialog** is used.

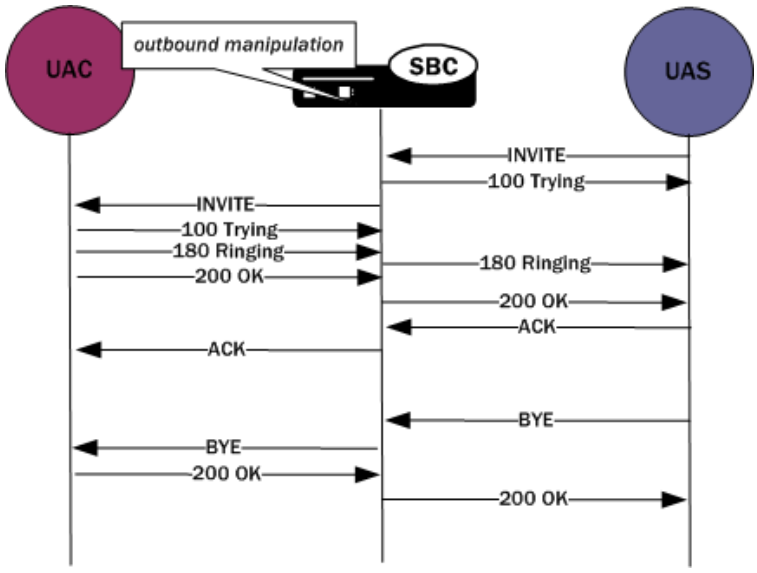
The Oracle Enterprise Communications Broker performs the restoration of original headers outside of SIP manipulations. That is, there are no manipulation rules to configure for restore the header to their original context. The Oracle Enterprise Communications Broker will recognize the headers have been modified, and restore them to their original state prior to sending the message out on the wire. Restoration takes place prior to outbound manipulations so that any outbound manipulation can those headers once they have been restored.



Outbound HMR Challenge

When you use the **out-of-dialog** setting for an outbound manipulation, the Oracle Enterprise Communications Broker only executes this specific SIP header rule only if the same SIP header rule was executed against the initial dialog-creating request.

For example, if the INVITE's To header was not manipulated, it would not be correct to manipulate the To header in the BYE request. To do so would render the UAC unable to properly match the dialog. And this also means that the outbound manipulation should be carried out against a To, From, or Call-ID header in the BYE request if it was manipulated in the INVITE.



Built-In SIP Manipulations

In the course of HMR use, certain rules have become commonly used. Lengthy and complex, these rules do not include any customer-specific information and do they can be used widely. To make using them easier, they have been turned into built-in rules that you can reference in the **in-manipulationid** and **out-manipulationid** parameters that are part of the realm, session agent, and SIP interfaces configurations.

Built-in rules start with the prefix ACME_, so Oracle recommends you name your own rules in a different manner to avoid conflict.

While the number of built-in manipulation rules is expected to grow, one is supported at the present time: ACME_NAT_TO_FROM_IP. When performed outbound, this rule changes:

- The To-URI hostname to the logical $TARGET_IP and port to $TARGET_PORT
- The From-URI to the logical $REPLY_IP and port to be $REPLY_PORT

Unique HMR Regex Patterns and Other Changes

In addition to the HMR support it offers, the Oracle Enterprise Communications Broker can now be provisioned with unique regex patterns for each logical remote entity. This supplement to pre-existing HMR functionality saves you provisioning time and saves Oracle Enterprise Communications Broker resources in instances when it was previously necessary to define a unique SIP manipulation per PBX for a small number of customer-specific rules.

Manipulation Pattern Per Remote Entity

On the Oracle Enterprise Communications Broker, you can configure logical remote entities (session agents, realms, and SIP interfaces) with a manipulation pattern string that the system uses as a regular expression. Then the SIP manipulation references this regular expression using the reserved word $MANIP_PATTERN. At runtime, the Oracle Enterprise Communications Broker looks for the logical entity configured with a manipulation pattern string in this order of preference: session agent, realm, and finally SIP interface.

On finding the logical entity configured with the manipulation string, the Oracle Enterprise Communications Broker dynamically determines the expression. When there is an invalid reference to a manipulation pattern, the pattern-rule expression that results will turn out to be the default expression (which is \,+).

When the $MANIP_PATTERN is used in a manipulation rule's **new-value** parameter, it resolves to an empty string, equivalent of no value. Even though this process ends with no value, it still consumes system resources. And so Oraclerecommends you do not use $MANIP_PATTERN as a **new-value** value.

In the following example, the SIP manipulation references the regular expression from a realm configuration:

```
realm-config
        identifier                              net200
        description
        addr-prefix                             0.0.0.0
        network-interfaces                      public:0
        ...
        manipulation-pattern                    Lorem(.+)
sip-manipulation
        name                            manip
        description
        header-rules
                name                            headerRule
                header-name                     Subject
                action                          manipulate
                match-value                     $MANIP_PATTERN
                msg-type                        request
                comparison-type                 pattern-rule
                new-value                       Math
                methods                         INVITE
```

Reject Action

When you use this action type and a condition matching the manipulation rule arises, the Oracle Enterprise Communications Broker rejects the request (though does not drop responses) and increments a counter.

- If the **msg-type** parameter is set to **any** and the message is a response, the Oracle Enterprise Communications Broker increments a counter to show the intention to reject the message—but the message will continue to be processed.
- If the **msg-type** parameter is set to **any** and the message is a request, the Oracle Enterprise Communications Broker performs the rejection and increments the counter.

The **new-value** parameter is designed to supply the status code and reason phrase corresponding to the reject. You can use the following syntax to supply this information: status-code[:reason-phrase]. You do not have to supply the status code and reason phrase information; by default, the system uses 400:Bad Request.

If you do supply this information, then the status code must be a positive integer between 300 and 699. The Oracle Enterprise Communications Broker then provides the reason phrase corresponding to the status code. And if there is no reason phrase, the system uses the one for the applicable reason class.

You can also customize a reason phrase. To do so, you enter the status code followed by a colon (:), being sure to enclose the entire entry in quotation marks () if your reason code includes spaces.

When the Oracle Enterprise Communications Broker performs the **reject** action, the current SIP manipulation stops processing and does not act on any of the rules following the **reject** rule. This course of action is true for nested SIP manipulations that might have been constructed using the **sip-manip** action type.

SNMP Support

The Oracle Enterprise Communications Broker provides SNMP support for the Rejected Messages data, so you can access this information externally. The new MIB objects are:

```
apSysRejectedMessages    OBJECT-TYPE
        SYNTAX          Counter32
        MAX-ACCESS      read-only
        STATUS          current
        DESCRIPTION
                "Number of messages rejected by the SD due to matching criteria."
        ::= { apSysMgmtMIBGeneralObjects 18 }
apSysMgmtRejectedMesagesThresholdExeededTrap        NOTIFICATION-TYPE
        OBJECTS         { apSysRejectedMessages }
        STATUS          current
        DESCRIPTION
        " The trap will be generated when the number of rejected messages exceed
the configured threshold within the configured window."
        ::= { apSystemManagementMonitors 57 }
apSysMgmtRejectedMessagesGroup  OBJECT-GROUP
        OBJECTS {
            apSysRejectedMessages
        }
        STATUS          current
        DESCRIPTION
                "Objects to track the number of messages rejected by the SD."
        ::= {  apSystemManagementGroups 18 }
apSysMgmtRejectedMessagesNotificationsGroup NOTIFICATION-GROUP
      NOTIFICATIONS {
                        apSysMgmtRejectedMesagesThresholdExeededTrap
                      }
                STATUS          current
                DESCRIPTION
                "Traps used for notification of rejected messages"
      ::= {  apSystemManagementNotificationsGroups 26 }
apSmgmtRejectedMessagesCap
            AGENT-CAPABILITIES
            PRODUCT-RELEASE     "Acme Packet SD"
            STATUS               current
            DESCRIPTION         "Acme Packet Agent Capability for enterprise
                                 system management MIB."
            SUPPORTS            APSYSMGMT-MIB
                INCLUDES        {
                                  apSysMgmtRejectedMessagesGroup,
                                  apSysMgmtRejectedMessagesNotificationsGroup
                                }
            ::= { apSmgmtMibCapabilities 37 }
```

Log Action

When you use this action type and a condition matching the manipulation rule arises, the Oracle Enterprise Communications Broker logs information about the current message to a separate log file. This log files will be located on the same core in which the SIP manipulation

occurred. On the core where sipt runs, a logfile called matched.log will appear when this action type is executed.

The matched.log file contains a timestamp, received and sent Oracle Enterprise Communications Broker network interface, sent or received IP address:port information, and the peer IP address:port information. It also specifies the rule that triggered the log action in this syntax: rule-type[rule:name]. The request URI, Contact header, To Header, and From header are also present.

```
---------------------------------------
Apr 17 14:17:54.526 On [0:0]192.168.1.84:5060 sent to 192.168.1.60:5060
element-rule[checkRURIPort]
INVITE sip:service@192.168.1.84:5060 SIP/2.0
From: sipp <sip:+2125551212@192.168.1.60:5060>;tag=3035SIPpTag001
To: sut <sip:service@192.168.1.84>
Contact: sip:sipp@192.168.1.60:5060
```

Name Restrictions for Manipulation Rules

Historically, you have been allowed to configure any value for the name parameter within a manipulation rule. This method of naming caused confusion when referencing rules, so now manipulation rules name must follow a specific syntax. They must match the expression ^[[alpha:]][[:alnum:]_]+$ and contain at least one lower case letter.

In other words, the name must:

- Start with a letter, and then it can contain any number of letters, numbers, or underscores
- Contain at least one lower case letter

All pre-existing configurations will continue to function normally. If you want to change a manipulation rule, however, you are required to change its name if it does not follow the new format.

The ACLI **verify-config** command warns you if the system has loaded a configuration containing illegal naming syntax.

Please note that the software allows you to make changes to HMRs, including configuring new functionality to existing rules, as long as you do not change the rule name. This results in an important consideration surrounding HMRs with hyphens in previously configured rule names.

- You can reference stored values in new value names. (Recall that stored values may be rule names.)
- You can perform subtraction in new value names.

If you use a rule names with hyphens within the REGEX of new value names, the system cannot determine whether the hyphen is part of the rule name or is intended to invoke subtraction within the REGEX. For this reason, you need to use great care with legacy HMR naming that includes hyphens.

As a general rule, create new rule names that follow the new rule naming guidelines if you intend to use new functionality in those rules.

New Value Restrictions

To simplify configuration and remove possible ambiguity, the use of boolean and equality operators (==, <=, <, etc.) for **new-value** parameter values has been banned. Since there was no

specific functionality tied to their use, their ceasing to be use will have no impact to normal SIP manipulation operations.

Header Manipulation Rules for SDP

The Oracle Enterprise Communications Broker supports SIP header and parameter manipulation rules for four types of SIP message contents:

- headers
- elements within headers
- ASCII-encoded Multipurpose Internet Mail Extensions (MIME) bodies
- binary-encoded MIME ISDN User Part (ISUP) bodies

While Session Description Protocol (SDP) offers and answers can be manipulated in a fashion similar to ASCII-encoded MIME, such manipulation is primitive in that it lacks the ability to operate at the SDP session- and media-levels.

In addition, the system supports a variant of Header Manipulation Rules (HMR) operating on ASCII-encoded SDP bodies, with specific element types for descriptors at both the session-level and media-level, and the ability to apply similar logic to SDP message parts as is done for SIP header elements.

The configuration object, mime-sdp-rules, under sip-manipulation specifically addresses the manipulation of SDP parts in SIP messages. Just as existing header-rules are used to manipulate specific headers of a SIP message, mime-sdp-rules will be used to manipulate the SDP specific mime-attachment of a SIP message.

SDP Manipulation

mime-sdp-rules function in a similar fashion as header-rules. They provide

- parameters used to match against specific SIP methods and/or message types
- parameters used to match and manipulate all or specified parts of an SDP offer or answer
- a means of comparing search strings or expressions against the entire SDP
- different action types to allow varying forms of manipulation

Since only a single SDP can exist within a SIP message, users need not specify a content-type parameter as is necessary for a mime-rule. A mime-sdp-rule operates on the single SDP within the SIP message. If no SDP exists with the message, one can be added. If the message already contains a mime attachment, adding SDP results in a multipart message.

All manipulations performed against all or parts of the SDP are treated as UTF-8 ASCII encoded text. At the parent-level (mime-sdp-rule) the **match-value** and **new-value** parameters execute against the entire SDP as a single string.

An add action only succeeds in the absence of SDP because a message is allowed only a single SDP offer or answer. A delete operation at the mime-sdp-rule level will remove the SDP entirely.

Note that on an inbound sip-manipulation, SDP manipulations interact with the Oracle Enterprise Communications Broker codec-policy. SDP manipulations also interact with codec reordering and media setup. It is very possible to make changes to the SDP such that the call can not be setup due to invalid media parameters, or settings that will affect the ability to

transcode the call. Consequently, user manipulation of the SDP can prove risky, and should be approached with appropriate caution.

Three configuration-objects, sdp-session-rule, sdp-media-rule, and mime-header-rule, exist under the mime-sdp-rule. These objects provide finer grained control of manipulating parts of the SDP.

sdp-session-rule

An sdp-session-rule groups all SDP descriptors, up until the first media line, into a single entity, thus allowing the user to perform manipulation operations on a session-specific portion of the SDP.

Like the mime-sdp-rule, all match-value and new-value operations performed at this level are executed against the entire session group as a complete string. Given the sample SDP below, if an sdp-session-rule is configured, the match-value and new-values operate only on the designated portion.

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait
```

Nested under the sdp-session-rule configuration object is an sdp-line-rule object, the object that identifies individual descriptors within the SDP. The types of descriptors used at the sdp-session-rule level are v, o, s, i, u, e, p, c, b, t, r, z, k, and a, the descriptors specific to the entire session description.

This level of granularity affords the user a very simple way to making subtle changes to the session portion of the SDP. For instance, it is very common to have to change the connection line at the session level.

The add and delete actions perform no operation at the sdp-session-rule level.

sdp-media-rule

An sdp-media-rule groups all of the descriptors that are associated with a specific media-type into single entity, thus allowing the user to perform manipulation operations on a media-specific portion of the SDP. For example, a user can construct an sdp-media-rule to change an attribute of the audio media type.

Like a mime-sdp-rule, all match-value and new-value operations performed at this level are executed against the entire media-group as a complete string. Given the sample SDP below, if a media-level-descriptor is configured to operate against the application group, the match-value and new-values would operate only on designated portion.

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
```

```
        u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
        e=mjh@isi.edu (Mark Handley)
        c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
        a=recvonly
        m=audio 49170 RTP/AVP 0
        m=video 51372 RTP/AVP 31
        m=application 32416 udp wb
        a=orient:portrait
```

A configuration parameter **media-type** is used to specify the media group on which to operate. It contains all of the descriptors including the m-line up to the next m-line. This parameter is a string field and must match the media-type exactly as it appears within the SDP. The special media-type media can be used to refer to all media types. This is particularly useful when performing an add operation, when the user wants to add a media section between the first and second medias, but does not know what media type they are. Otherwise, during an add operation, the media section would be added before the specified media-type (if no index parameter was provided).

The types of descriptors used at the sdp-media-rule level are m, i, c, b, k, and a, the descriptors specific to the media description.

This level of granularity affords the user a very simple way to making subtle changes to the media portion of the SDP. For instance, it is very common to have to change the name of an audio format (for example G729 converted to g729b), or to add attributes specific to a certain media-type.

The index operator is supported for the media-type parameter (for example, media-type audio[1]). Like header rules, if no index is supplied, this means operate on all media-types that match the given name. For specifying specific media-types, the non-discrete indices are also supported (for example, ^ - last). Adding a media-type, without any index supplied indicates that the media should be added at the beginning. The special media-type media uses the index as an absolute index to all media sections, while a specific media-type will index relative to that given media type.

For sdp-media-rules set to an action of add where the media-type is set to media, the actual media type is obtained from the new-value, or more specifically, the string after m= and before the first space.

Given the following SDP:

```
        v=0
        o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
        c=IN IP4 224.2.17.12/127
        t=2873397496 2873404696
        m=audio 49170 RTP/AVP 0
        m=audio 48324 RTP/AVP 8
        m=video 51372 RTP/AVP 31
```

With the sdp-media-rule:

```
sdp-media-rule
        name                        smr
        media-type                  audio[1]
        action                      manipulate
        comparison-type             case-sensitive
        match-value
        new-value                   "m=audio 1234 RTP/AVP 8 16"
```

This rule operates on the 2nd audio line, changing the port and adding another codec, resulting in the SDP:

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
m=audio 49170 RTP/AVP 0
m=audio 1234 RTP/AVP 8 16
m=video 51372 RTP/AVP 31
```

The following rule, however:

```
sdp-media-rule
        name                    smr
        media-type              media[1]
        action                  add
        comparison-type         case-sensitive
        match-value
        new-value               "m=video 1234 RTP/AVP 45"
```

adds a new video media-type at the 2nd position of all media-lines, resulting in the SDP:

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
m=audio 49170 RTP/AVP 0
m=video 1234 RTP/AVP 45
m=audio 48324 RTP/AVP 8
m=video 51372 RTP/AVP 31
```

sdp-line-rule

Unlike header-rules, sdp descriptors are not added in the order in which they are configured. Instead they are added to the SDP adhering to the grammar defined by RFC 4566 (as is shown below).

```
Session description
   v=  (protocol version)
   o=  (originator and session identifier)
   s=  (session name)
   i=* (session information)
   u=* (URI of description)
   e=* (email address)
   p=* (phone number)
      c=* (connection information -- not required if included in
        all media)
   b=* (zero or more bandwidth information lines)
   One or more time descriptions ("t=" and "r=" lines; see
        below)
   z=* (time zone adjustments)
   k=* (encryption key)
   a=* (zero or more session attribute lines)
   Zero or more media descriptions (see below)

Time description
   t=  (time the session is active)
   r=* (zero or more repeat times)

Media description, if present
```

```
m=  (media name and transport address)
i=* (media title)
c=* (connection information -- optional if included at
     session level)
b=* (zero or more bandwidth information lines)
k=* (encryption key)
a=* (zero or more media attribute lines)
```

* after the equal sign denotes an optional descriptor.

This hierarchy is enforced meaning that if you configure a rule which adds a session name descriptor followed by a rule which adds a version descriptor, the SDP will be created with the version descriptor first, followed by the session name.

The only validation that will occur is the prevention of adding duplicate values. In much the same way that header-rules prevents the user from adding multiple To headers, the descriptor rule will not allow the user to add multiple descriptors; unless multiple descriptors are allowed, as is in the case of b, t, r and a.

There exists a parameter **type** under the sdp-line-rule object that allows the user to specify the specific line on which to perform the operation. For example: v, o, s, i, u, e, p, c, b, t, r, z, k, a, and m. Details on these types can be found in RFC 4566.

For those descriptors, of which there may exist zero or more (b, t, r, and a) entries, indexing grammar may be used to reference the specific instance of that attribute. This indexing grammar is consistent with that of header-rules for referring to multiple headers of the same type.

Given the example SDP below:

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
r=604800 3600 0 90000
r=7d 1h 0 25h
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait
```

and the following sdp-line-rule:

```
sdp-line-rule
   name                 removeRepeatInterval
   type                 r[1]
   action               delete
```

The rule removeRepeatInterval removes the second repeat interval descriptor within the SDP.

The behavior of all SDP rules follow the same behavior of all manipulation rules in that they are executed in the order in which they are configured and that each rule executes on the resultant of the previous rule.

Each descriptor follows its own grammar and rules depending on the type specified. The values of the descriptor are evaluated at runtime since the new-values themselves are evaluated at

runtime. At this time no validation of the grammar for each of the types is performed. The user is responsible for properly formatting each of the descriptors according to their specifications.

For instance, the version (v) descriptor can be removed from the SDP but leaving all descriptors for that SDP, causing the SDP to become invalid. This is consistent with the way header-rules operate, in that there is no validation for the specific headers once they have been manipulated through HMR.

Regular Expression Interpolation

An interpolated regular expression is a regular expression that is compiled and evaluated at runtime. Today all regular expressions are compiled at configuration time in order to improve performance. There are cases where a regular expression is determined dynamically from data within a SIP message. In these circumstances the regular expression is unknown until the time of execution.

In order to have a regular expression be interpolated at runtime, it must be contained within a set of {}. An interpolated expression can have any number of regular expressions and strings appended together. Any characters to the left or right of the curly braces will be appended to the value within the curly braces. The curly braces are effectively two operators treated as one (interpolate the value contained within and then concatenate the values to the left and right of the curly braces). If the comparison-type is set to pattern-rule and the match-value contains a value that matches the grammar below, then it will be treated as an interpolated expression.

([^\\]|^)\{\$[^0-9]+[^}]*\}

The example below demonstrates using a user defined variable within a regular expression of another rule at runtime.

```
element-rule

        name                            someRule
        type                            header-value
        action                          replace
        comparison-type                 pattern-rule
        match-value                     ^sip:{$rule1.$0}@(.+)$
        new-value                       sip:bob@company.com
```

If the value of $rule1.$0 evaluates to alice then it will successfully match against the string sip:alice@comcast.net. An interpolated expression can be as simple as “{$rule1.$0}” or as complex as ^sip:{rule1.$0}@{$rule2[1].$2}$. It is not possible to interpolate a normal regular expression since the grammar will not allow the user to enter such an expression. Only variables can be contained with the curly braces.

The resultant of interpolated expressions can be stored in user defined variables. Given the same example from above, if the rule someRule was referenced by another rule, the value of sip:alice@comcast.net would be stored within that rule.

Interpolation only makes a single pass at interpolation, but does so every time the Rule executes. In other words, if the Rule is applied to the Route header, it will interpolate again for each Route header instance. What this means is that the value within the curly braces will only be evaluated once. For instance, if the value {$someRule.$1} evaluates to {$foobar.$2} the Oracle Enterprise Communications Broker (ECB) will treat $foobar.$2 as a literal string which it will compile as a regular expression. The ECB will not recursively attempt to evaluate $foobar.$2, even if it was a valid user defined variable.

Interpolated regular expressions will evaluate to TRUE if an only if both the regular expression itself can be compiled and it successfully matches against the compared string.

Regular Expressions as Boolean Expressions

Regular expressions can be used as boolean expressions today if they are the only value being compared against a string, as is shown in the case below.

```
mime-rule
        name                            someMimeRule
        content-type                    application/text
        action                          replace
        comparison-type                 pattern-rule
        match-value                     ^every good boy .*
        new-value                       every good girl does fine
However, regular expressions can not be used in conjunction with other boolean
expressions to form more complex boolean expressions, as is shown below.

mime-rule
     name                       someMimeRule
     content-type               application/text
     action                     replace
     comparison-type            boolean
     match-value                $someRule & ^every good boy .*
     new-value                  every good girl does fine
```

There are many cases where the user has the need to compare some value as a regular expression in conjunction with another stored value. It is possible to perform this behavior today, however it requires an extra step in first storing the value with the regular expression, followed by another Manipulation Rule which compares the two boolean expressions together (e.g. $someRule & $someMimeRule).

In order to simplify the configuration of some sip-manipulations and to make them more efficient this functionality is being added.

Unfortunately, it is not possible to just use the example as is shown above. The problem is there are many characters that are commonly used in regular expressions that would confuse the HMR expression parser (such as $, and +). Therefore delimiting characters need to be used to separate the regular expression from the other parts of the expression.

To treat a regular expression as a boolean expression, it needs to be enclosed within the value $REGEX(<expression>,<compare_string>=$ORIGINAL); where <expression> is the regular expression to be evaluated. <compare_string> is the string to compare against the regular expression. This second argument to the function is defaulted to $ORIGINAL which is the value of the of the specific Manipulation Rule object. It can be overridden to be any other value the user desires.

The proper configuration for the example above to use regular expressions as boolean expressions is

```
mime-rule
        name             someMimeRule
        content-type     application/text
        action           replace
        comparison-type  boolean
        match-value      $someRule & $REGEX(“^every good boy .*”)
        new-value        every good girl does fine
```

It is also possible to use expressions as arguments to the $REGEX function. These expressions will in turn be evaluated prior to executing the $REGEX function. A more complex example is illustrated below.

```
header-rule
        name              checkPAU
        header-name       request-uri
        action            reject
        comparison-type   boolean
        match-value       (!$REGEX($rule1[0],$FROM_USER))&
                          (!$REGEX($rule2[0],$PAI_USER))
        msg-type          request
        new-value         403:Forbidden
        methods           INVITE,SUBSCRIBE,MESSAGE,PUBLISH,
                          OPTIONS, REFER
```

It should be noted that when using $REGEX() in a boolean expression, the result of that expression is not stored in the user variable. The comparison-type must be set to pattern-rule in order to store the result of a regular expression.

The arguments to the $REGEX() function are interpolated by default. This is the case since the arguments themselves must be evaluated at runtime. The following example is also valid.

```
mime-rule
        name               someMimeRule
        content-type       application/text
        action             replace
        comparison-type    boolean
        match-value        $someRule & $REGEX(“^every good
                           {$rule1[0].$0} .*”)
```

Moving Manipulation Rules

Users can move rules within any manipulation-rule container. Any manipulation rule which contains sub-rules will now offer the ACLI command **move** <from index> <to index>. For example, given the order and list of rules below:

1. rule1

2. rule2

3. rule3

4. rule4

Moving rule3 to position 1 can be achieved by executing **move 3 1**. The resulting order will then be: rule3, rule1, rule2, rule4. A move operation causes a shift (or insert before) for all other rules. If a rule from the top or middle moves to the bottom, all rules above the bottom are shifted up to the position of the rule that was moved. If a rule from the bottom or middle moves to the top, all rules below are shifted down up to the position of the rule that was moved. Positions start from 1.

A valid from-index and to-index are required to be supplied as arguments to the move action. If a user enters a range that is out of bounds for either the from-index or to-index, the ACLI will inform the user that the command failed to execute and for what reason.

With respect to the issue of creating an invalid sip-manipulation by incorrectly ordering the manipulation rules, this issue is handled by the Oracle Enterprise Communications Broker validating the rules at configuration time and treating them as invalid prior to runtime. This may or may not affect the outcome of the sip-manipulation as a configured rule may not perform any operation if it refers to a rule that has yet to be executed. It is now the user’s responsibility to reorder the remaining rules in order to make the sip-manipulation valid once again.

It is important to note that rules of a different type at the same level are all part of the same list. To clarify; header-rules, mime-rules, mime-isup-rules and mime-sdp-rules all share the same configuration level under sip-manipulation. When selecting a move from-index and to-index for a header-rule, one must take into consideration the location of all other rules at the same level, since the move is relative to all rules at that level, and not relative to the particular rule you have selected (for example, the header-rule).

Since the list of rules at any one level can be lengthy, the **move** command can be issued one argument at a time, providing the user with the ability to select indices. For instance, typing **move** without any arguments will present the user with the list of all the rules at that level. After selecting an appropriate index, the user is then prompted with a to-index location based on the same list provided.

For Example:

```
ORACLE(sip-mime-sdp-rules)# move
select a rule to move
```

1: msr sdp-type=any; action=none; match-value=; msg-type=any

2: addFoo header-name=Foo; action=none; match-value=; msg-type=any

3: addBar header-name=Bar; action=none; match-value=; msg-type=any

```
selection: 2
destination: 1
Rule moved from position 2 to position 1
ACMEPACKET(sip-mime-sdp-rules)#
```

Rule Nesting and Management

There will be cases where the user wants to take a stored value from the SDP and place it in a SIP header, and vice-versa. All header-rules, element-rules, mime-rules, mime-isup-rules, isup-param-rules, mime-header-rules and mime-sdp-rules are inherited from a Manipulation Rule. A Sip Manipulation is of type Manipulation which contains a list of Manipulation Rules. Each Manipulation Rule can itself contain a list of Manipulation Rules. Therefore when configuring manipulation rules, they will be saved in the order which they have been configured. This is different from the way other configuration objects are configured. Essentially, the user has the option of configuring which type of object they want and when they are done, it gets added to the end of the sip-manipulation, such that order is preserved. This will mean that any Manipulation Rule at the same level can not share the same name. For example, names of header-rules can't be the same as any of the mime-sdp-rule ones or mime-isup-rule. This allows the user to reference stored values from one rule type in another at the same level.

ACLI Configuration Examples

The following eight sections provide sample SDP manipulations.

Remove SDP

```
sip-manipulation
        name                    stripSdp
        description             remove SDP from SIP message
        mime-sdp-rule
                name                    sdpStrip
                msg-type                request
                methods                 INVITE
```

```
                action                  delete
                comparison-type         case-sensitive
                match-value
                new-value
```

Remove Video from SDP

```
sip-manipulation
        name                        stripVideo
        description                 strip video codecs from SIP
                                    message
        mime-sdp-rule
                name                    stripVideo
                msg-type                request
                methods                 INVITE
                action                  manipulate
                comparison-type         case-sensitive
                match-value
                new-value
                sdp-media-rule
                        name                removeVideo
                        media-type          video
                        action              delete
                        comparison-type     case-sensitive
match-value
                        new-value
```

Add SDP

```
sip-manipulation
        name                      addSdp
        description               add an entire SDP if one does
                                  not exist
        mime-sdp-rule
                name                    addSdp
                msg-type                request
                methods                 INVITE
                action                  add
                comparison-type         case-sensitive
                match-value
                new-value               “v=0\r\no=mhandley
2890844526 2890842807 IN IP4 “+$LOCAL_IP+”\r\ns=SDP Seminar\r\ni=A
Seminar on the session description protocol\r\nu=http:
//www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps\r\ne=mjh@isi.edu
(Mark Handley)\r\nc=IN IP4 “+$LOCAL_IP+”\r\nt=2873397496
2873404696\r\na=recvonly\r\nm=audio 49170 RTP/AVP 0\r\n”
```

Manipulate Contacts

This rule changes the contact in the SDP to the value contained in the Contact header.

```
sip-manipulation
        name                            changeSdpContact
        description                     changes the contact in the SDP to the
value of the contact header
        header-rule
                name                            storeContact
                header-name                     Contact
                action                          store
                comparison-type                 pattern-rule
```

```
                msg-type                        request
                methods                         INVITE
                match-value
                new-value
                element-rule
                        name                            storeHost
                        parameter-name
                        type                            uri-host
                        action                          store
                        match-val-type                  ip
                        comparison-type                 pattern-rule
                        match-value
                        new-value
        mime-sdp-rule
                name                            changeConnection
                msg-type                        request
                methods                         INVITE
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                new-value
                sdp-session-rule
                        name                    changeCLine
                        action                  manipulate
                        comparison-type         case-sensitive
                        match-value
                        new-value
                        sdp-line-rule
                           name                 updateConnection
                           type                 c
                           action               replace
                           comparison-type      case-sensitive
                           match-value          $storeContact.$storeHost
                           new-value            $storeContact.$storeHost.$0
```

Remove a Codec

This rule changes the contact in the SDP to the value contained in the Contact header.

```
sip-manipulation
        name                            removeCodec
        description                     remove G711 codec if it exists
        mime-sdp-rule
                name                            removeCodec
                msg-type                        request
                methods                         INVITE
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                new-value
                sdp-media-rule
                        name                        removeG711
                        media-type                  audio
                        action                      manipulate
                        comparison-type             case-sensitive
                        match-value
                        new-value
                        sdp-line-rule
                                name                    remove711
                                type                    m
                                action                  replace
```

```
                                comparison-type         pattern-rule
                                match-value             ^(audio [0-9]
                                                        {1,5} RTP.*)( [07]
                                                        \b)(.*)$
                                new-value               $1+$3
                        sdp-line-rule
                                name                    stripAttr
                                type                    a
                                action                  delete
                                comparison-type         pattern-rule
                                match-value             ^(rtpmap|fmtp):
                                                        [07]\b$
                                new-value
```

Change Codec

```
sip-manipulation
        name                            convertCodec
        description                     changeG711toG729
        mime-sdp-rule
                name                            changeCodec
                msg-type                        request
                methods                         INVITE
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                new-value
                sdp-media-rule
                        name                      change711to729
                        media-type                audio
                        action                    manipulate
                        comparison-type           case-sensitive
                        match-value
                        new-value
                        sdp-line-rule
                                name                  change711
                                type                  m
                                action                replace
                                comparison-type       pattern-rule
                                match-value           ^(audio [0-9]{4,5}
                                                      RTP/AVP.*)( 0)(.*)$
new-value             $1+" 18"+$3
                        sdp-line-rule
                                name                  stripAttr
                                type                  a
                                action                delete
                                comparison-type       pattern-rule
                                match-value           ^rtpmap:0 PCMU/
                                                      .+$
                                new-value
                        sdp-line-rule
                                name                  addAttr
                                type                  a
                                action                add
                                comparison-type       boolean
                                match-value           $change711to729.
                                                      $stripAttr
                                new-value             rtpmap:18 G729/8000
```

Remove Last Codec and Change Port

```
sip-manipulation
        name                            removeLastCodec
        description                     remove the last codec
        mime-sdp-rule
                name                            removeLastCodec
                msg-type                        request
                methods                         INVITE
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                new-value
                sdp-media-rule
                        name                        removeLast
                        media-type                  audio
                        action                      manipulate
                        comparison-type             case-sensitive
                        match-value
                        new-value
                        sdp-line-rule
                                name                    isLastCodec
                                type                    m
                                action                  store
                                comparison-type         pattern-rule
                                match-value             ^(audio )([0-9]{4,
                                                        5})( RTP/AVP
                                                        [0-9]{1-3})$
new-value
                        sdp-line-rule
                                name                    changePort
                                type                    m
                                action                  replace
                                comparison-type         boolean
                                match-value             $removeLastCodec.
$removeLast.$isLastCodec
                                new-value               $removeLastCodec.
$removeLast.$isLastCodec.$1+0+$removeLastCodec.$removeLast.
$isLastCodec.$3
```

Remove Codec with Dynamic Payload

```
sip-manipulation
        name                            removeAMR
        description                     remove the AMR and AMR-WB dynamic codecs
        mime-sdp-rule
                name                            sdpAMR
                msg-type                        request
                methods                         INVITE
                action                          manipulate
                comparison-type                 case-sensitive
                match-value
                new-value
                sdp-media-rule
name                      mediaAMR
                        media-type                  audio
                        action                      manipulate
                        comparison-type             case-sensitive
                        match-value
```

```
                new-value
                sdp-line-rule
                        name              isAMR
                        type              a
                        action            delete
                        comparison-type   pattern-rule
                        match-value       ^rtpmap:([0-9]
                                          {2,3}) AMR
                        new-value
        sdp-media-rule
                name                      mediaIsAMR
                media-type                audio
                action                    manipulate
                comparison-type           boolean
                match-value               $sdpAMR.$media
                                          AMR.$isAMR
                new-value
                sdp-line-rule
                        name              delFmtpAMR
                        type              a
                        action            delete
                        comparison-type   pattern-rule
                        match-value       ^fmtp:{$sdpAMR.
                                          $mediaAMR.
                                          $isAMR.$1}\b
                        new-value
                sdp-line-rule
                       name             delAMRcodec
                       type             m
                       action           find-replace-all
                        comparison-type pattern-rule
                        match-value     ^audio [0-9]+
                                        RTP.*( {$sdpAMR.
                                         $mediaAMR.$isAMR.
```

HMR Import-Export

Due to the complexity of SIP manipulations rules and the deep understanding of system syntax they require, it is often difficult to configure reliable rules. This feature provides support for importing and exporting pieces of SIP manipulation configuration in a reliable way so that they can be reused.

To Import HMRs, use the **Upload** link on the sip-manipulation list dialog, which is the first dialog displayed after clicking the HMR icon. To export, use **Download**.

Exporting

The SIP manipulation configuration contains an **export** command which sends the previously selected configuration to the designated file. The syntax is **export [FILENAME]**. The system compresses the file with gzip and writes it to the `/code/imports` directory.

> **Note:**
>
> SIP manipulation configurations can only be exported one at a time.

Exported data will look like this:

```
<?xml version='1.0' standalone='yes'?>
<sipManipulation
        name='manip'
        description=''
        lastModifiedBy='admin@console'
        lastModifiedDate='2009-10-16 14:16:29'>
        <headerRule
                headerName='Foo'
                msgType='any'
                name='headerRule'
                action='manipulate'
                cmpType='boolean'
                matchValue='$REGEX("[bB][A-Za-z]{2}")'
                newValue='foo'
                methods='INVITE'>
        </headerRule>
</sipManipulation>
```

To avoid conflicts when importing, the key and object ID are not included as part of the exported XML.

Importing

The **import** command imports data from a previously exported file into the currently-selected configuration. If no configuration was selected, a new one is created. The syntax is **import [FILENAME]**. Include the .gz extension in the filename. After importing, type **done** to save the configuration.

Importing a configuration with the same key as one that already exists returns an error. In this case:

- Delete the object with the same key and re-import.
- Select the object with the same key and perform an import that will overwrite it with new data.

Using SFTP to Move Files

After exporting a configuration, use SFTP to copy the file to other Oracle Enterprise Communications Brokers. Place the file in the `/code/imports` directory before using the **import** command on the second ECB.