BEA WebLogic SIP Server®

Release Notes

Version 2.2
Revised: May 16, 2006

Copyright

Restricted Rights Legend

Trademarks and Service Marks

Contents

WebLogic SIP Server 2.2 Features and Changes

Welcome to BEA WebLogic SIP Server 2.2! WebLogic SIP Server™ integrates SIP Servlet technologies with J2EE 1.3 and other leading Internet standards to provide a reliable framework for developing highly available, scalable, and secure telecommunications applications. WebLogic SIP Server's seamless integration of disparate, heterogeneous platforms and applications enables your network to leverage existing software investments and share the enterprise-class services and data that are crucial to building next-generation telephony applications.

The following sections describe the new features and changes made in the WebLogic Server 2.2 general release and in intermediate releases:

- "What's New in WebLogic SIP Server 2.2?" on page 1-1
- "What's New in WebLogic SIP Server 2.1?" on page 1-9
- "What's New in WebLogic SIP Server 2.0 SP2" on page 1-13
- "Deprecated Features in WebLogic SIP Server 2.0 SP1" on page 1-13

What's New in WebLogic SIP Server 2.2?

This section describes new features and functionality introduced in WebLogic SIP Server 2.2.

New RFC Support

WebLogic SIP Server 2.2 now supports the following additional RFCs and specifications:

- RFC 2543: SIP: Session Initiation Protocol (v1) backward compatibility is supported.

- RFC 3262: Reliability of Provisional Responses in the Session Initiation Protocol (SIP) is supported.
- RFC 3311: The Session Initiation Protocol (SIP) UPDATE Method
- RFC 3327: Session Initiation Protocol (SIP) Extension Header Field for Registering Non-Adjacent Contacts is supported.
- RFC 3515: The Session Initiation Protocol (SIP) Refer Method is supported.

See Supported Specifications and RFCs for a full description of RFC compliance.

Changes for 3GPP Application Server Compliance

WebLogic SIP Server 2.2 introduces a variety of changes in order to comply better with the IMS Application Server specifications described in 3GPP 24.229 Rel 7.0.0:

- The 3GPP specification states that applications that interact with a S-CSCF should add an `orig` parameter when creating new requests based on an identity specified in the top route header. WebLogic SIP Server provides a new `com.bea.wcp.sip.util.TransportUtil` class to help an application determine if a particular request was generated locally, so that the application can add the `orig` parameter if necessary. See the WebLogic SIP Server JavaDoc.
- The SIP container now supports messages having the `Request-Disposition`, `Accept-Contact`, and `Reject-Contact` headers, which can be used by callers to define preferences for how a message should be processed. See RFC 3840.
- When acting as a Proxy, WebLogic SIP Server 2.2 ensures that no more than one `P-Charging-Vector` or `P-Charging-Function-Address` header is present in a given message, and automatically strips any such header if a message is proxied to an untrusted host. See RFC 3455.
- WebLogic SIP Server 2.2 uses 3GPP security processing workflows for SIP and HTTP authentication.
- The `sipserver.xml` file includes a new element, `route-header` which you can use to statically define the S-CSCF route header to be added to new outgoing requests originated from the WebLogic SIP Server container. See route-header in the *Configuration Reference*.
- The UPDATE method described in RFC 3311 is now supported. See "Support for SIP UPDATE Method (RFC3311)" on page 1-3.
- An extended API is provided to help applications modify Contact header parameters. See "Support for Modifying Contact Header Parameters" on page 1-8.

Diameter Sh Interface, Relay Node Support, and Profile Service API

WebLogic SIP Server implements the Diameter Sh interface as a Web Application that you can deploy to the server. Sh interface functions are implemented as a provider that transparently generates and responds to the Diameter command codes defined in the Sh application specification. A higher-level profile service API enables SIP Servlets to manage user profile data as an XML document using XML Document Object Model (DOM). WebLogic SIP Server supports converged SIP and Diameter applications. See Using the Profile Service API (Diameter Sh Interface) in *Developing Applications with WebLogic SIP Server*.

WebLogic SIP Server also provides a Diameter relay node application, which you can configure for use when connecting to an HSS. An HSS simulator application is included for development or testing purposes. See Configuring Diameter Sh Client Nodes and Relay Agents in *Configuring and Managing WebLogic SIP Server*.

Support for SIP UPDATE Method (RFC3311)

As described in RFC 3311, the SIP UPDATE method enables a UAC to update the parameters of a session, such as the media streams or codecs used for communication. To fully support the behavior described in RFC 3311, WebLogic SIP Server now automatically generates a 500 response with a `Retry-After` header value if a deployed B2BUA receives an UPDATE request, relinquishes control (returns control from the `service()` method), and then subsequently receives another UPDATE. This change was made to comply with section 5.2 of RFC 3311, which requires a UAS to return a 500 response and `Retry-After` header if a second UPDATE is received before making a final response to a prior UPDATE.

Path Header Support (RFC3327)

RFC 3327 defines the extension header field, Path, which provides a mechanism for multiple SIP proxies to add themselves to a defined path between a UAC and registrar. The Path header functions similar to the SIP Record-Route header, but is defined outside of a Dialog.

WebLogic SIP Server provides API support to enable multiple proxy applications to add themselves to a Path header in a REGISTER request. A deployed registrar application could then manage and maintain the Path headers.

To provide this support, WebLogic SIP Server extends the `javax.servlet.sip.Proxy` interface with `com.bea.wcp.sip.WlssProxy`. The extended interface provides getter/setter methods for the `AddToPath` attribute, which determines whether `proxyTo()` operations automatically add a

Path header to the proxied request. The interface also provides a method to retrieve the complete URI defined in the Path header for a request, so that the proxy can add arbitrary attributes to the header.

See the com.bea.wcp.sip.WlssProxy JavaDoc for more information.

Note that WebLogic SIP Server does not provide a method for a proxy application to push an arbitrary Path entry (as well as its own Path as supported in the extended API).

Support for SIP REFER Method (RFC3515)

WebLogic SIP Server supports the SIP refer-to header and REFER method for applications that implement services such as unattended or attended call transfer, and third-party call control.

A “REFER” request can come within an existing dialog, or out of dialog. If REFER arrives out of a dialog, it starts a new dialog. WebLogic SIP Server creates an implicit subscription after the 202 Accepted response is sent, and the dialog remains active until the subscription expires. The subscription and original dialog are maintained until a subsequent NOTIFY or method explicitly terminates the subscription. The subscription for REFER can also be terminated with a SUBSCRIBE request sent with expiration = 0.

If a NOTIFY request doesn't find a REFER subscription it is rejected with a 481 status response.:

Because of the implicit subscription described above, WebLogic SIP Server maintains a SIP dialog after the REFER method is used even if a subsequent BYE message is sent. This behavior ensures that call transfer applications can re-establish the original session (via an INVITE) if the call transfer fails. See Section 2.4 Transfer - Unattended in http://tools.ietf.org/wg/sipping/draft-ietf-sipping-service-examples/draft-ietf-sipping-service-examples-09.txt for more information.

WebLogic SIP Server extends the JSR 116 `SipServlet` class to provide a convenience method for handling SIP REFER requests. To provide a handler your own SIP Servlet, extend `com.bea.wcp.sip.WlssSipServlet` and implement the `doRefer()` method. For example:

```
package myapp;

import javax.io.IOException;
import javax.servlet.ServletException;
import com.bea.wcp.sip.WlssSipServlet;

public class MyApp extends WlssSipServlet {

  protected void doRefer(SipServletRequest req)
    throws ServletException, IOException {

  // Respond to REFER method...
```

```
    }
}
```

Changes to SIP Request Authentication for 3GPP TS 24.229

The authentication process for SIP requests having the P-Asserted-Identity header was changed to conform to the behavior described in 3GPP TS 24.229: IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP). In version 2.2, if the SIP message contains a Privacy header with either an “id” or “user” value, the user is treated as being anonymous. Previously, only the “id” value was considered. See Configuring P-Asserted-Identity Assertion in *Configuring Security for WebLogic SIP Server*.

WebLogic SIP Server 2.2 authentication behavior meets the requirements of 3GPP TS 24.229 except as follows:

- The server performs authentication only for protected resources. To fully conform to 3GPP TS 24.229, SIP Servlets should use declarative security to assign permissions to all resources. See Securing SIP Servlet Resources in *Developing Applications with WebLogic SIP Server*.
- If an anonymous user fails an authorization check, WebLogic SIP Server forces authentication by challenging the user using the `auth-method` configured in the `login-config` element of the application’s `sip.xml` descriptor. This is done instead of immediately returning a 403 Forbidden response.
- The server does not evaluate the From header when handling messages with P-Asserted-Identity. This behavior is not necessary because the server automatically forces authentication by challenging the user if an anonymous user fails an authorization check.
- WebLogic SIP Server does not enforce a maximum number of authentication challenges. Instead, a user is locked for a period of time if repeated authentication attempts fail. See Setting Lockout Attributes for User Accounts in the WebLogic Server 8.1 documentation for more information.
- 3GPP TS 24.229 specifies that a server should provide the option to return a 2xx final response for a non-anonymous user when the user is not authorized to access a requested resource. WebLogic SIP Server does not provide the option to return these “fake” 2xx responses, and instead issues a 403 Forbidden response. If necessary, application code can easily construct “fake” 2xx responses using one of two different method:
  – **Using programmatic security rather than declarative security.** An Application can use the `HttpServletRequest.isUserInRole()` method to determine if a user is

authorized to access a particular resource. If the user is not authorized, the application can construct a "fake" 2xx response to return to the UAC and mask the authorization failure.

– **Using application composition.** One Servlet in an application chain can act as a filter that converts 403 Forbidden responses to a 2xx final response.

Support for X-3GPP-Asserted-Identity Header (3GPP TS 33.222)

WebLogic SIP Server now supports the `X-3GPP-Asserted-Identity` header for HTTP authentication as specified in 3GPP TS 33.222. Two new security providers, `X3gppAssertedIdentityAsserter` or `X3gppAssertedIdentityStrictAsserter` are available to enable and configure support. See Configuring 3GPP HTTP Authentication Providers in *Configuring Security for WebLogic SIP Server*.

Converged Application Support

WebLogic SIP Server now supports the development and deployment of *converged applications* that combine SIP protocol functionality with features from J2EE such as HTTP sessions, Enterprise JavaBeans (EJBs), and Java Message Service (JMS).

An extended API is provided to help you obtain `SipApplicationSession` objects and to create and associate HTTP sessions with `SipApplicationSession` objects.

An extended API is also provided to help non-SIP Servlets concurrently modify a `SipApplicationSession` object in a replicated environment. See Developing Converged Applications in *Developing Applications with WebLogic SIP Server*. See also the converged application example installed with WebLogic SIP Server (`*wlss_home*/samples/server/examples/src/convergence/readme.html`).

Production Application Upgrade

You can upgrade a deployed SIP application to a different version without losing calls that the application is current processing. WebLogic SIP Server supports application upgrades by allowing two different versions of the same application to be deployed simultaneously. The server itself automatically routes new calls to the latest-deployed application version, while directing messages for already-established calls to the older application version. After a period of time, the new application version processes all SIP messages, while the older version processes no messages and can be safely undeployed. See Upgrading Deployed SIP Applications in *Configuring and Managing WebLogic SIP Server*.

Improved Failover Detection

A new failover detection mechanism is provided to improve failover performance in the event that a data tier server process immediately fails, or the network connection between an engine and data tier server is physically compromised. See Improving Failover Performance for Physical Network Failures in *Configuring and Managing WebLogic SIP Server*.

Content Indirection

WebLogic SIP Server now supports SIP applications that indirectly specify the content of the SIP message body (for example, using an HTTP URL). Content indirection is generally used move large message bodies out of the SIP signalling network, or to allow bandwidth-constrained applications to determine whether or not they should download the additional content. Instead of providing the content directly in the message, the message body contains only a link to the specified content and metadata that specifies the size and type of information to be retrieved.

To support content indirection, WebLogic SIP Server provides an API to enable SIP Servlets to easily determine whether or not a message uses content indirection. For messages that include indirect content, the API provides for easy retrieval of the content metadata. See Using Content Indirection in SIP Servlets in *Developing Applications with WebLogic SIP Server*. See also the draft specification for content indirection at http://tools.ietf.org/wg/sip/draft-ietf-sip-content-indirect-mech/draft-ietf-sip-content-indirect-mech-05.txt.

Reliable Provisional Responses (RFC 3262)

WebLogic SIP Server supports reliable provisional responses (defined in http://www.ietf.org/rfc/rfc3262.txt). For a particular dialog there may be multiple reliable provisional responses at a given time. For this reason, the existing `SipSession.createRequest()` method in the SIP Servlet API is not sufficient for handling PRACK. Also, the basic `doAck()` method cannot be used for generating PRACK from the reliable provisional responses themselves.

Because of these issues, WebLogic SIP Server provides two alternatives for creating PRACK from the reliable provisional responses themselves:

1. `WlssSipServletResponse.createPrack()`—SIP Servlets can acknowledge a provisional response (and stop retransmissions of provisional responses) by creating a PRACK request using the `com.bea.wcp.sip.WlssSipServletResponse.createPrack()` method. This method returns a PRACK request having the RAck header sequence number of the response that is being acknowledged.

2. `SipServletResponse.doAck()`—The implementation of the JSR116 `SipServletResponse.createAck()` method was modified to automatically return a PRACK request instead of an ACK request for provisional (non-2xx) responses. `doAck()` is allowed only for reliable provisional responses.

See the WebLogic SIP Server JavaDoc for `com.bea.wcp.sip.WlssSipServletResponse`.

Support for Modifying Contact Header Parameters

WebLogic SIP Server provides limited support for modifying Contact header parameters, which is required by the 3GPP IMS specification. In version 2.2, applications can use the following `SipServletMessage` method call to return a Contact address object whose parameters can be modified:

```
SipServletMessage.getAddressHeader(“Contact”);
```

The object returned allows only limited modification of parameters in this release. Specifically, the object does not permit applications to add a URI user part to the Contact header, or to modify parameters that could influence the network interface that the SIP Container uses for transmitting messages.

In addition to the above change, the code was modified to allow a UAS to modify Contact header parameters for a 200 response to an OPTIONS request, as required in RFC 3261.

New Example Applications

WebLogic SIP Server 2.2 introduces a number of new and revised example applications. Source and build files for the new examples are installed at *`WLSS_HOME`*`\samples\server\examples\src`, where *`WLSS_HOME`* is the directory in which you installed WebLogic SIP Server (for example, `c:\bea\wlss220`).

See the *`WLSS_HOME`*`\samples\server\examples\src\index.html` file for information about all examples installed with WebLogic SIP Server.

Support for Generating SNMP Traps from SIP Servlets

WebLogic SIP Server 2.2 introduces a new runtime MBean, `SipServletSnmpTrapRuntimeMBean`, that enables applications to easily generate SNMP traps. The WebLogic SIP Server MIB contains seven new OIDs that are reserved for traps generated by an application. See Generating SNMP Traps from Application Code in *Developing Applications with WebLogic SIP Server*.

Default SIP Servlet Configuration

A new configuration element in `sipserver.xml`, `default-servlet-name`, enables you to specify a default SIP Servlet that the container calls for incoming initial requests when no other Servlet can be matched via `servlet-mapping` definitions in `sip.xml`. See default-servlet-name in the *Configuration Reference*.

The `default-servlet-name` element cannot be modified using the Administration Console. You must modify this value directly in `sipserver.xml` using a text editor, and redeploy the `sipserver/config` Web Application to all engine tier servers, to use this feature. You can redeploy the `config` Web Application using either the Administration Console or the `weblogic.Deployer` utility. To redeploy from the Administration Console:

1. For single-server domains, expand the Servers node and select the name of the WebLogic SIP Server instance. For multiple-server domains, expand the Clusters node and select the name of the engine tier cluster.
2. Select the Deployments->Web Modules tab in the right pane.
3. Select the config application from the table.
4. Select the Deploy tab.
5. Click Redeploy next to the single server instance, or the engine tier cluster on which the application is deployed.

To redeploy using the `weblogic.Deployer` utility, enter a command to:

```
java java weblogic.Deployer -username weblogic -password weblogic -verbose
   -targets config@BEA_ENGINE_TIER_CLUST -name sipserver -adminurl
   t3://localhost:7001 -redeploy"
```

What's New in WebLogic SIP Server 2.1?

This section details features that were introduced between WebLogic SIP Server 2.1 and earlier versions. This section includes information about the following:

- "Architectural Changes" on page 1-10
- "Application Porting Guidelines" on page 1-10
- "New Security Features" on page 1-10
- "Configuration Changes" on page 1-11

- "Container Changes for send() Calls" on page 1-12

Architectural Changes

The architecture of WebLogic SIP Server 2.1 was dramatically improved to provide increased performance, higher availability, and flexibility in configuring available resources. The most visible change in architecture is in WebLogic SIP Server's use of two separate clusters, referred to as the engine tier and data tier, that can be sized independently of one another to increase the call throughput or availability of an installation. (Development installations and small production installations can also use a single server instance as needed.) See Overview of the WebLogic SIP Server Architecture for more information about these changes. See also Capacity Planning for WebLogic SIP Server Deployments for information about sizing each cluster to match the performance and reliability goals of your applications.

WARNING: When you configure a domain with multiple engine and data tier servers, you must accurately synchronize all server system clocks to a common time source (to within one or two milliseconds) in order for the SIP protocol stack to function properly. See Configuring NTP for Accurate SIP Timers for more information.

Application Porting Guidelines

SIP Servlets developed for previous versions of WebLogic SIP Server must observe new coding practices and requirements in order to operate in the version 2.1 distributed environment:

- Applications should not create threads. When the `do`*`xxx`* method of a SIP Servlet is invoked by the SIP Servlet container, the container automatically locks the associated call state. If the `do`*`xxx`* method spawns additional threads or accesses a different call state, deadlock scenarios can occur.
- All Servlets should be non-blocking.
- All Session data must be serializable.
- Servlets must use the `distributable` tag, in order to deploy them to a cluster of engine tier WebLogic SIP Server instances. The `distributable` tag is not required and is ignored if you deploy to a single combined-tier (non-replicated) WebLogic SIP Server instance.

These and other porting requirements are described in Requirements and Best Practices for WebLogic SIP Server Applications in *Developing Applications with WebLogic SIP Server*.

New Security Features

WebLogic SIP Server now supports the `P-Asserted-Identity` SIP header as described in RFC3325. See Trusted Host Forwarding with P-Asserted-Identity in *Configuring and Managing WebLogic SIP Server*.

WebLogic SIP Server now supports Client-Cert authentication as well as BASIC and Digest authentication. See Configuring Client-Cert-Based Authentication for SIP Applications in *Configuring and Managing WebLogic SIP Server*. Client-Cert authentication is disabled by default; the switch to enable is defined in `ClusterMBean` and `ServerMBean`.

WebLogic SIP Server 2.1 now includes an RDBMS (JDBC) Digest Identity Assertion provider as well as the LDAP provider included in previous versions. In addition, both the RDBMS and LDAP providers support reverse-encrypted passwords as well as clear-text and hashed password values. See Configuring Digest Authentication for more information.

Configuration Changes

The following sections describe changes in the way you configure and manage WebLogic SIP Server 2.1.

datatier.xml Changes (Formerly statetier.xml)

The configuration file used to define partitions and replicas in the data tier is now named `datatier.xml`. In addition, the main XML element defined in the file has changed to `data-tier` (formerly `state-tier`). The location of the data tier configuration file has also changed; both `datatier.xml` and `statetier.xml` are located in the *`DOMAIN_DIR`*`/sipserver/config` subdirectory, where *`DOMAIN_DIR`* is the root directory of the WebLogic SIP Server domain.

Load Balancer Configuration Changes

Load balancer addresses are no longer defined in the `sipserver.xml` configuration file. Instead, the load balancer address and port number must be defined in the **External Listen Address** and **External Listen Port** fields of a SIP channel on each engine tier server. See Configuring Load Balancer Addresses in *Configuring and Managing WebLogic SIP Server*.

Changes in Queue Length-Based Overload Protection

When using queue- length-based overload protection controls, WebLogic SIP Server now monitors the *sum* of the lengths of the `sip.transport.Default` and `sip.timer.Default` execute queues, rather than only the length of `sip.transport.Default`. Also, the default overload configuration initiates overload protection when the combined queue size reaches 200 simultaneous requests, and releases overload control when the combined size falls to 150 simultaneous requests. See overload in *Configuring and Managing WebLogic SIP Server*.

sipserver.xml Changes

The schema for `sipserver.xml` has changed in WebLogic SIP Server. See the Engine Tier Configuration Reference (sipserver.xml) in *Configuring and Managing WebLogic SIP Server*. Notable changes include:

- Proxy entries now use single-elements that include a full URI (rather than multiple elements for each portion of a URI).
- The terminology and XML elements for regulating incoming SIP calls has changed. See overload.

In addition to schema changes, the location of `sipserver.xml` has changed in this release. The `sipserver.xml` is included as part of the `sipserver` enterprise application that implements the SIP container features of WebLogic SIP Server. See Overview of WebLogic SIP Server Configuration and Management.

Network Configuration Using Channels

WebLogic SIP Server 2.1 no longer uses the (previously-deprecated) `connector` element in `sipserver.xml` for configuring network connections. Instead, all connections are defined using:

- The listen address and listen port for the WebLogic SIP Server instance, and
- All network channel resources configured for the WebLogic SIP Server instance.

Multiple network channels can be defined to support multiple transport protocols or to configure multiple network interfaces on multi-homed server hardware. See the Managing WebLogic SIP Server Network Resources in *Configuring and Managing WebLogic SIP Server*.

Access Logging Configuration Changes

Access logging is no longer configured by defining a filter in the `sipserver.xml` configuration file. See Enabling Access Logging in *Developing Applications with WebLogic SIP Server* for information about the new XML configuration elements.

Container Changes for send() Calls

In previous WebLogic SIP Server releases, if an application called the `send()` method within a SIP request method such as `doInvite()`, `doAck()`, `doNotify()`, and so forth, the SIP Servlet container immediately transmitted the `send()` call to the network. In WebLogic SIP Server 2.1, `send()` calls are instead buffered in the order in which they are called, and transmitted in order only after the control has returned to the SIP Servlet container.

WARNING: Applications must not wait or sleep after a call to `send()`, because the request or response is not transmitted until control returns to the SIP Servlet container.

What's New in WebLogic SIP Server 2.0 SP2

WebLogic SIP Server 2.0.2 introduced the following features:

- Digest Authentication support was introduced using a new LDAP Digest Authentication Identity Asserter and a separate authorization provider. See Configuring Digest Authentication for WebLogic SIP Server.
- Deployment Descriptors in `weblogic.xml` introduced support for mapping principal and role names to roles defined in `sip.xml`. See Securing SIP Servlet Resources.
- SNMP support was enabled by default; a patch was no longer required.
- Operator documentation for WebLogic SIP Server SNMP Traps was made available. See Understanding and Responding to SNMP Traps.
- Administrator documentation was made available to help configure Administration Servers in a WebLogic SIP Server domain. See Administration Server Best Practices in *Managing and Monitoring WebLogic SIP Server*.
- Developer documentation was made available to describe how to use the Eclipse IDE to develop SIP Servlets with WebLogic SIP Server. See Developing SIP Servlets Using Eclipse.

Deprecated Features in WebLogic SIP Server 2.0 SP1

WebLogic SIP Server 2.0.1 was a restricted release of a BEA product and was subject to change in future releases. The following features were specifically called out as having been deprecated in WebLogic SIP Server 2.0.1:

- Load balancer URIs will be obtained from Network Channel attributes, instead of `sipserver.xml`, in a future release of WebLogic SIP Server.

- Future releases of WebLogic SIP Server will not use the built-in WLSSecurityManagerFilter and custom security manager to implement digest authentication.
- `sipserver.xml` may not be used to configure SIP Servlet container features in future releases.
- The session backup implementation was deprecated and subject to change in future releases.
- Creating general-purpose SIP filters by extending a base filter class was deprecated in this release.

WebLogic SIP Server 2.2 Known Issues

The following table summarizes known issues and problems in WebLogic SIP Server 2.2.

Note: This section describes only those issues associated with the SIP Servlet container and data replication features of WebLogic SIP Server 2.2. See also the WebLogic Server 8.1 Known Issues for information about known problems with WebLogic Server 8.1, which provides the underlying OA&M and J2EE capabilities of WebLogic SIP Server 2.2.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
n/a	By default, new Diameter network channels are created with a default Idle Connection Timeout value of 65 seconds. Change this attribute from the default in order to ensure that connections are not dropped and recreated every 65 seconds. See Creating Network Channels for the Diameter Protocol in *Configuring and Managing WebLogic SIP Server*.
n/a	WebLogic SIP Server MIB objects are read-only. You cannot modify a WebLogic SIP Server configuration using SNMP.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
n/a	This version of Weblogic SIP Server exhibits two behaviors that do not conform to the JSR 116 specification: ● MIME content is returned as a String object, rather than as a `javax.mail.Multipart` object as encouraged by the specification. ● `isPersistent`, used for re-instantiating `ServletTimer` after server restarts, is not implemented. Also, WebLogic SIP Server does not support dialog stateless proxies, an optional feature described in the API JavaDoc for the `Proxy` interface, `setStateful()` method: "This proxy parameter is a hint only. Implementations may choose to maintain transaction state regardless of the value of this flag, but if so the application will not be invoked again for this transaction."
n/a	If you attempt to install WebLogic SIP Server 2.2 on Fedora Core 3 or 4 with `selinux` running, the installer throws a `java.lang.UnsatisfiedLinkError` exception. You cannot install WebLogic SIP Server while `selinux` is active.
n/a	If you configure two or more data tier replicas using the default WebLogic Server Listen Address configuration (which specifies no listen address), multiple data tier instances on the same machine cannot connect to one another. This problem occurs because, using the default Listen Address configuration, JNDI objects in the first booted server bind to all local IP addresses. To avoid this problem, always enter a valid IP address for each configured data tier server instance.
n/a	In a WebLogic SIP Server installation with two engine tier nodes and two data tier nodes in a partition (two replicas), if the connection to the data tier becomes "split" such that each engine tier server can only reach a different data tier node, one of the replicas is forced offline. To recover from this situation, always configure the Node Manager utility to restart data tier replicas automatically when a replica fails. This enables the replica to rejoin its associated partition and update its copy of the call state data without having to manually restart the server.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
n/a	When using a replicated WebLogic SIP Server installation, the default execute queue configuration for data tier servers poses a risk that garbage collection pauses in engine tier servers will cause delays in servicing other engine tier servers. To minimize this risk, on each data tier server set the thread count for the weblogic.kernel.Default queue to twice the number of engine tier servers in your deployment. You can set the thread count in the Administration Console by following these instructions: 1. Expand the Servers tab in the left pane. 2. Right-click the name of a data tier server and select View Execute Queues. 3. Click the weblogic.kernel.Default queue in the right pane. 4. Change the Thread Count attribute to equal twice the number of engine tier servers in y our system. 5. Click Apply. 6. Repeat the above instructions for each data tier server. Increasing the thread count in this manner minimizes the risk that garbage collection pauses in an engine tier server will delay service to other engine tier servers in the data tier.
CR244201	The Administration Console lists the `sipserver` implementation application as a standard J2EE application, and allows a Console user to redeploy or even remove the application from a running WebLogic SIP Server installation. The `sipserver` application must never be undeployed or redeployed except indirectly via the `ConfigManagerRuntimeMBean`. Redeploying the application yields several nested exceptions starting with `InstanceAlreadyExistsException`, and forces running data tier server instances to shut down. To avoid these problems, never redeploy or undeploy the `sipserver` application using the Administration Console or `weblogic.Deployer` utility. Perform all engine tier configuration changes using the SIP Servers node in the Console or the WLST command-line utility, as described in Configuring Engine Tier Container Properties.
CR252501	In the Administration Console, the Monitoring->General tab displays “Undefined” for the Active Application Session Count and Active SIP Session Count attributes when monitoring a replicated WebLogic SIP Server deployment. There is currently no workaround for this problem.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
CR267829	When starting a replicated domain, if a partition has no running replicas and two replicas are started at the same time, the second replica shuts down if one or more engine tier servers are already running. To avoid this problem, always start all data tier servers ***before*** starting any engine tier servers in a replicated domain.
CR272491, CR189353	On Linux and UNIX systems, the default TCP connection timeout interval is usually very long and can cause Managed Servers to disconnect from the Administration Server under certain failure conditions. Specifically, if a single Managed Server in a domain fails abruptly or is disconnected from the network (for example, due to a removed network cable), the Administration Server tries to communicate to the failed server for the length of the TCP connection timeout value. During this time, the Administration Server does not send heartbeat messages to the remaining Managed Servers in the domain. Failing to send the heartbeat messages causes the remaining Managed Servers to consider the Administration Server as being offline, and they disconnect from the Administration Server. This finally causes the Administration Server to throw `PeerGoneException`s for the disconnected servers after the TCP timeout interval has been reached and the connection is closed. To work around this issue without changing the operating system TCP connection timeout value, use the `-Dweblogic.client.SocketConnectTimeoutInSecs` startup option when booting the Administration Server. BEA recommends using a value of 60 seconds to avoid numerous missed heartbeats (`-Dweblogic.client.SocketConnectTimeoutInSecs=60`).
CR273935	WebLogic Server 8.5 Service Pack 5 cannot interoperate with OpenSSL when acting as a client. When WebLogic Server initiates a connection, OpenSSL shuts down the connection upon handshake, and no TLS error is generated. There is currently no workaround to this problem.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description

CR276039	The Diameter Sh client application included in WebLogic SIP Server 2.2 uses threads from the `sip.transport.Default` execute queue. Because this queue is also used for general SIP message processing, applications that use the Sh interface may experience poor performance with the default execute queue settings. To work around this problem, increase the number of threads available in the `sip.transport.Default` queue to a large number (for example, 200 threads).
	To change the queue length, perform these steps for each engine tier server:
	1. Access the Administration Console for your domain.
	2. Expand the Servers node.
	3. Right-click the name of an engine tier server, and select View Execute Queues.
	4. Select `sip.transport.Default` in the list of queues in the table.
	5. Change the **Thread Count** value to the desired number of threads (for example, 200).
	6. Click Apply and reboot the server.
	The Sh client application may also consume additional threads in `sip.transport.Default` if the HSS is unavailable. This problem occurs because the Sh application uses a large default timeout value (30 seconds) when waiting for a response from the HSS. Using a smaller `timeout` value (for example, 1 second) ensures that available threads are not quickly consumed when the HSS is unavailable.
	To change the `timeout` value, edit the `diameter.xml` configuration file for engine tier servers to configure the `timeout` parameter. For example:
	`...`
	`<application>`
	`  <auth-application-id>16777217</auth-application-id>`
	`  <vendor-id>10415</vendor-id>`
	`  <class-name>com.bea.wcp.diameter.sh.WlssShApplication</`
	`class-name>`
	`  <param>`
	`    <name>timeout</name>`
	`    <value>1000</value>`
	`  </param>`
	`</application>`
	`...`

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
CR276602	Although the `sipserver.xsd` schema defines the element, `engine-call-state-cache-enabled`, this feature is not supported in WebLogic SIP Server 2.2

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description
CR276062, CR276897	Beginning with WebLogic SIP Server 2.2, the "replicated" domain deploys the `sipserver` implementation application using the default "stage" mode, rather than the "nostage" mode used in previous releases. With stage mode deployment, if you manually edit a configuration file (`sipserver.xml`, `datatier.xml`, or `diameter.xml`), you must explicitly redeploy the `config` Web Application component in `sipserver` to all target server instances. See the instructions under "Default SIP Servlet Configuration" on page 1-9. Stage mode deployment also changes the procedure for applying patches to WebLogic SIP Server. After applying a patch as described in Applying Patches Using InstallPatch in Configuring and Managing WebLogic SIP Server, you must either: • Manually delete the contents of the staging directory for each Managed Server (by default, *`domain_directory`*`/`*`server_name`*`/stage`) to force a refresh of the deployment files. • Manually copy the patch file into the `sipserver/APP-INF/container` directory in each server's stage directory.

Table 2-1 Version 2.2 Known Issues

Change Request Number	Description

CR277059	In the profile service API, the `SipApplicationSessionAdapter` is not recreated after a call state has been restored. This means that if a Servlet on a particular engine creates a profile service subscription and the server subsequently fails, another engine tier server that recreates the necessary call state cannot obtain the session with `ProfileSubscription.getApplicationSession()`. Attempting to recreate the session throws:
	`java.lang.AssertionError`
	`        at`
	`test.ProfileServlet.update(ProfileServlet.java:100)`
	`        at`
	`com.bea.wcp.sip.engine.server.CanaryServlet.update(CanaryServlet.java:1013)`
	`        at`
	`com.bea.wcp.diameter.sh.Subscription.notifyListener(Subscription.java:116)`
	`        at`
	`com.bea.wcp.diameter.sh.WlssShApplication$1.run(WlssShApplication.java:106)`
	`        at`
	`com.bea.wcp.sip.bea.wls81.connector.WLSTask.execute(WLSTask.java:43)`
	`        at`
	`weblogic.kernel.ExecuteThread.execute(ExecuteThread.java:224)`
	`        at`
	`weblogic.kernel.ExecuteThread.run(ExecuteThread.java:183)`

Resolved Problems in the WebLogic SIP Server 2.2

The following table summarizes the issues that were resolved in WebLogic SIP Server 2.2.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description
CR222494	The SIP Servlet container did not support `ejb-link` and `resource-ref` entries defined in the `sip.xml` deployment descriptor file. Instead the values had to be defined in `weblogic.xml` as a workaround. The code was modified to support these entries directly in `sip.xml`.
CR235377	Call overload controls were not enabled by default. This problem was address with a code fix.
CR236024	WebLogic SIP Server sometimes threw a NullPointerException when running a User Agent Client (UAC) against a proxy servlet that proxied back to the same engine tier server instance. The problem caused the exception: `<Client timer task failed with fatal status java.lang.NullPointerException at com.bea.wcp.sip.engine.server.SipServletMessageImpl.getDialogId(SipServletMessageImpl.java:274)` The problem was solved with a code fix.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description
CR236379	Configuration files used inconsistent naming conventions for the data tier and replicas within the data tier. The configuration file schema has changed to consistently use the term "data tier" to refer to the cluster of WebLogic SIP Server instances that manage call state data, "partition" to refer to a managed portion of the call state, and "replica" to refer to an individual WebLogic SIP Server instance within a partition. See Configuring Data Tier Partitions and Replicas and Data Tier Configuration Reference (datatier.xml).
CR236479	The SNMP MIB for WebLogic SIP Server was previously available only from Managed Servers running in a domain. The code was modified to make WebLogic SIP Server MIB entries available from the Administration Server as well as Managed Servers. See Configuring SNMP.
CR237487	WebLogic SIP Server did not listen for UDP messages on a non-default network channel that specified IP_ANY/0.0.0.0 as the listen address. The code was modified so that the server listens for incoming UDP messages on any IP interface when you define a network channel with 0.0.0.0 as the listen address. See Configuring Servers to Listen on Any IP Interface (0.0.0.0) in *Configuring and Managing WebLogic SIP Server*.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description

CR238527	When using the Hostpot 1.4.2_05 VM and running under heavy loads, the UDP NIO socket would sometimes fail with:
	`java.io.IOException: Interrupted system call`
	`        at`
	`sun.nio.ch.PollArrayWrapper.poll0(Native Method)`
	`        at`
	`sun.nio.ch.PollArrayWrapper.poll(PollArrayWrapper.java:100)`
	`        at`
	`sun.nio.ch.PollSelectorImpl.doSelect(PollSelectorImpl.java:64)`
	`        at`
	`sun.nio.ch.SelectorImpl.lockAndDoSelect(SelectorImpl.java:59)`
	`        at`
	`sun.nio.ch.SelectorImpl.select(SelectorImpl.java:70)`
	`        at`
	`com.bea.wcp.sip.engine.connector.transport.UdpTransportModule.run(UdpTransportModule.java:413)`
	The recovery from this failure caused the failure to lose 10 seconds of network traffic. This problem was resolved with a code fix.

CR239030	The previously deprecated XML configuration elements for defining trusted hosts have been replaced with new configuration elements. See sip-security in *Configuring and Managing WebLogic SIP Server*.
CR239032	Previously the **Tcp Connect Timeout Millis** attribute applied only to SIP protocol channels. The timeout setting was ignored for channels configured for the SIPS protocol. This problem was resolved with a code fix.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description

CR239250	In a replicated environment, or in a single server environment with debugging turned on, adding sleep time at the end of a `doMessage()` call could result in the error:
	`<Error> <WLSS.Session> <BEA-331410> <Invalid CSeq header. request=NOTIFY`
	`sip:8005551212@172.17.24.251;appsessionid=app-nnw8zlr1voya:840a17c2649c84d0a47f06f1d7062cd2%40172.17.24.251;pxxx=12341234,`
	`CSeq header=1 NOTIFY, CSeq number in this dialog=1>`
	This problem was resolved with a code fix.

CR240087	When waiting for over 60 minutes between an INVITE and a BYE message, a load testing proxy application would sent a 481 response even though the call should not be stateful. For example:
	`2005-08-22 14:12:51: Aborting call on unexpected message for Call-ID`
	`'1-8415@10.32.4.213': while expecting '200' response, received 'SIP/2.0 481`
	`Call/Transaction Does Not Exist`
	`To: testuser <sip:proxy@10.32.4.213:5060>;tag=1`
	`Content-Length: 0`
	`CSeq: 2 BYE`
	`Call-ID: 1-8415@10.32.4.213`
	`Via: SIP/2.0/UDP 10.32.4.213:5061;branch=z9hG4bK-1-6`
	`From: userb <sip:userb@10.32.4.213:5061>;tag=1`
	`Server: BEA WebLogic SIP Server 2.2.0.0`
	This problem was resolved by adding a new container configuration parameter, `default-behavior`, which defines wether WebLogic SIP Server should act as a proxy or a user agent (UA) in the absence of an available, matching application. See default-behavior in *Configuring and Managing WebLogic SIP Server*.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description
CR240670	Prior to version 2.2, a WebLogic SIP Server engine tier server would start up even if no SIP network channels were targeted to the server (for example, if a new engine tier server was configured manually and no channels were created). The code was changed so that engine tier servers now throw an exception and fail to start if no SIP channels have been configured for the server. The new error message is: `<Error> <WLSS.Engine> <BEA-330075> <There are no sip channels targeted to server "`*`servername`*`">`
CR241600	The previous version of the `findme` example application did not work in a domain having multiple engine tier servers in a cluster. The example code and documentation were modified to support a clustered environment. See Build the Example.

CR243700	WebLogic SIP Server did not persist session attributes after a Servlet made a call to `setAttribute()`. For example, in the following code sample the call to `modifyState()` did not persist call state data in the data tier:
	
	`Foo foo = new Foo(..);`
	
	`appSession.setAttribute("name", foo); // This persists the call state.`
	
	`foo.modifyState(); // This change is not persisted.`
	
	This problem was resolved with a code fix.

CR244502	The Administration Console allowed you to uncheck the **Outbound Enabled** attribute for a SIP or SIPS network channel, even though SIP and SIPS network channels can always originate outbound connections. In addition, the Console allowed you to select the **HTTP Enabled for This Protocol** attribute for SIP and SIPS channels even though HTTP and SIP/SIPS are not supported on the same port number. The Console code was modified to make these attributes read-only for SIP and SIPS network channels.

Table 3-1 Problems Resolved in Version 2.2

Change Request Number	Description
CR245393	The WebLogic Server Administration Console had several problems that could affect the configuration of WebLogic SIP Server: • CR241785: The Console did not prevent a user from assigning null to attributes that require actual values. For example, when configuring the Digest authentication provider, the Console would persist null to the mandatory `DigestRealmName` attribute if no value was specified, even though the server would fail to start with this configuration. • CR241822: The Console did not prevent a user from configuring multiple identity asserter providers that had the same active token type. • CR241825: When you viewed the configuration page for the digest identity asserter provider, the Console always showed `PLAINTEXT` as the configured value in the drop-down menu for the Password Encryption Type attribute. `PLAINTEXT` was displayed even if you had previously configured the provider with an alternate encryption type, such as `REVERSIBLEENCRYPTED` or `PRECALCULATEDHASH`. The code was modified to address these problems.
CR253622	If you defined a `message-debug` element with the `level` set to "full" and you also specified the `-Dwlss.SipEngine` debugging option at startup, the server failed to start with a NullPointerException. The code was modified to address this problem.

Resolved Problems for Service Pack 2

The following table summarizes the issues that were resolved in WebLogic SIP Server 2.0.2.

Table 4-1 Problems Resolved in Version 2.0.2

Change Request Number	Description
CR211125	The product license file, sip-license.xml, was moved to the WebLogic SIP Server product directory (`BEA_HOME/wlss202`).
CR217316	The code was modified so that SIP message Via, Contact, and Record-Route headers can be populated with the correct IP address on multihomed machines. See Setting Up Connectors.
CR218114	WebLogic SIP Server generated an exception if an INVITE request contained a Route header having the server's IP address. The code was modified to remove the Route header and forward the request.
CR218136	If a SIP request did not contain a Max-Forwards header, WebLogic SIP Server would throw a javax.servlet.sip.TooManyHopsException and respond with code 483. This behavior did not match the specified behavior, which is to decrement the Max-Forwards value by one of the header is present. The code was changed to match the specified behavior.
CR218285	WebLogic SIP Server threw an exception if a From address contained a "<" or ">" symbol. The code was modified to address this problem.

Table 4-1 Problems Resolved in Version 2.0.2

Change Request Number	Description
CR218359	The code was modified to support configurable SIP timers. See Configuring SIP Timers.
CR219912, CR221880	WebLogic SIP Server did not observe the Record-Route header for BYE or ACK messages. Instead, BYE and ACK messages were sent using the Contact header, ignoring the record-route hierarchy. This problem was solved with a change to the code.
CR221952	WebLogic SIP Server did not generate an error if it could not bind to a configured UDP port (for example, if the port was already in use). The code was modified to generate an appropriate message when the server cannot bind to a configured port.
CR224690	To improve proxy performance, the code was modified to first use UDP with a fixed buffer size, and then switch to TCP only if the message size exceeds the MTU size. This code fix also improves compliance with the SIP specification, because responses that exceed the configured MTU size are no longer rejected, but are instead sent over UDP if the original Request was via UDP.
CR231206	When forwarding messages having Route header values and a proxyTo() destination, WebLogic SIP Server forwarded to the proxyTo() destination rather than the first Route value. The code was changed to comply with the SIP specification; Route headers are now handled based on the transaction user in question and on whether a dialog is established.
CR231208	WebLogic SIP Server could potentially proxy a CANCEL request if a SIP application failed to implement a doCancel() method but the doRequest() method proxied the CANCEL. The code was modified to ensure that CANCEL requests are not proxied even if the SIP application attempts to proxy them in doRequest().
CR231821	WebLogic SIP Server returned an ArrayIndexOutOfBounds exception when 0 bytes were read from a stream. This issue was resolved with a code fix.

Table 4-1 Problems Resolved in Version 2.0.2

Change Request Number	Description
CR231849	WebLogic SIP Server 2.0.2 includes a “no-op” authentication provider, called the Identity Assertion Authenticator, that performs neither group population nor user existence checking. You can configure this provider and use it with the Digest Identity Asserter provider when neither group population nor user existence checking is required, in order to save an additional round-trip connection to the LDAP server. See Configuring Digest Authentication for WebLogic SIP Server for more information.
CR231852	Trusted hosts can now be configured in order to bypass authentication for listed host addresses. See Configuring Trusted Hosts.
CR231857	A deadlock situation could occur when two messages were received by a B2BUA servlet at the same time and each leg tried to forward the response to the other leg. This issue was resolved with a code fix.
CR231887	Weblogic SIP Server did not consider DNS names in Route headers when proxying requests; if a Route header contained a DNS name that resolved to the IP address of WebLogic SIP Server itself, the Route header was not removed when proxying the request to its destination. The code was changed to resolve the DNS name in the Route header and remove the header if the name resolved to the IP address of the server. WebLogic SIP Server caches the IP address to avoid repeated DNS lookups.