# RADVISION SIP Server Platform version 3.5 Release Notes
**March 2008**

This document contains late breaking or other information that supplements the SIP Server Platform version 3.5 documentation.

## Contents

## 1. Features

The RADVISION SIP Server Toolkit provides the functionality necessary for building SIP-aware server applications. The SIP Server Toolkit is a RFC 3261 standards-compliant library with the following add-on modules:

- Advanced Back-to-Back Application Framework (B2BAF) Add-on Module for writing back-to-back SIP applications.
- Events Server Add-on Module that includes specific support for Presence, Winfo, Publish and Presence Agent.
- LDAP Add-on Module for using LDAP Server for security and location services.
- Session Description Protocol (SDP) Add-on Module for session description protocol message management.
- Accounting Add-on Module for collection and dissemination of accounting-related data.

The SIP Server Platform also provides a High Availability (HA) Framework for integrating the SIP Server to a SIP Server cluster that provides a highly-available, high-performance and scalable virtual server.

Back to Top

## 2. New in this Version

### SIP Server Core

- DLA (Dynamic Local Address)—dynamically changes the listen address of the server. This feature allows developers to dynamically change the Via and Record-Route information, as well as the list of domains that the server is managing. This feature facilitates the development of servers that frequently have listen address changes, and allows changing the listen address and domain list on the fly with no need to restart.
- Content-ID header support for the Content-ID header was added. The Content-ID header is now identified as the RVSIP_HEADERTYPE_CONTENT_ID header, and can be manipulated through the various RvSipContentIDHeader.h API functions. The Content-ID value may be used for uniquely identifying MIME entities in different contexts.

### Back-to-Back Application Framework

- Termination Session Timer support—manages session timers of each termination independently. This capability extends the SIP Server 3.0 capability of relying Session Timer related messages between the UAC and the UAS. This feature allows the development of server B2B applications that independently maintain the Session Timer session of the different B2B call-legs. This feature facilitates the development of servers, such as Inter-work functions (IWF) that translate between capabilities of different clients.
- Store and Restore HA API —stores and restores the run-time state of the service. The Store HA API allows obtaining a buffer which contains the serialized data of a service state. Restore allows restoring a service object given a serialized data buffer. This API is important for customers who wish to store and restore the service run-time state using their proprietary HA framework. This feature facilitates developing servers which offer redundancy and call continuity.
- 3rd Party Call Control (3PCC) Disconnect Service—disconnects a call-leg. This service abstracts the low level intricacies of the state machine that handles the disconnect procedure of a call-leg. Using this service, developers can command the call-leg to disconnect gracefully without being aware of the call-leg state. This service aids in performing fine control actions on call sessions, such as disconnecting a single leg of a multi-point conference service.
- B2BAFClient Service reference implementation—provides a reference implementation for a service that hides the low-level control flow of a call-leg. This service abstracts the low-level intricacies of the state machine of a call-leg for the automatic handling of call-leg session timers.

- API for child enumeration-enumerates child services of a specific service. This API allows developing advanced services that need to send events to different child services in the B2BAF Service Tree. This feature facilitates in the development of advanced services, such as Mid-Call Announcement services, and so on.

## IMS P-Headers

- P-Profile-Key header support for the P-Profile-Key header was added. The P-Profile-Key header is now identified as the RVSIP_HEADERTYPE_P_PROFILE_KEY header, and can be manipulated through the various RvSipPProfileKeyHeader.h API functions. This header is no longer handled as an Other header.
- P-User-Database header-support for the P-User-Database header was added. The P-User-Database header is now identified as the RVSIP_HEADERTYPE_P_USER_DATABASE header, and can be manipulated through the various RvSipPUserDatabaseHeader.h API functions. This header is no longer handled as an Other header.
- Additional p-Headers—two new headers are now supported: the P-Early-Media header (implemented as the Other header) and the P-Answer-State header. Also, additions to the P-Access-Network-Info header are in the access-type parameter values.

## Platform Support and Development Environment

- 64bit support for Windows and Linux OS.
- Support for Microsoft Visual Studio 2005 IDE.

Back to Top

## 3. Supported Operating Systems

The following table details the availability of OS-related features across operating systems and OS version support. The current release is available on Windows, Solaris and Linux. For detailed information regarding OS versions, see the *RADVISION SIP Platform version 3.5 Operating Systems Specific Information Readme File*.

| Operating System | Version | IPv6 | TLS | Multi-threaded |
|---|---|---|---|---|
| Solaris | 9 | Yes | Yes | Yes |
| | 10 | Yes | Yes | Yes |
| Linux Red Hat | Red Hat Enterprise 3 | Yes | Yes | Yes |
| | Red Hat Enterprise 4 | Yes | Yes | Yes |
| | Red Hat Enterprise 5 | No | Yes | Yes |
| | RedHat Enterprise 64 bit (processor types: ia64, x86_64) | Yes | Yes | Yes |
| Windows | Windows 2003 Server | Yes | Yes | Yes |
| | 2003 64 bit | Yes | No | Yes |
| | XP | Yes | Yes | Yes |

Back to Top

## 4. What this Package Contains

The package contains the source code for the SIP Server Platform, including the SIP Stack. The package also contains extensive documentation (see Documentation Information) and different test and sample applications as described below:

- **GUI Test application for non-embedded operating systems**—a ready-to-use program that demonstrates how to implement a SIP Server of different types: Proxy (stateless or transaction stateful), Registrar, Redirect Server, B2BAF, Accounting, and Events Server.

  The Test Application provides a rich set of features, such as SIP Server configuration, diverse object control, and resource monitoring. The Test Application also demonstrates use of a remote LDAP Server. The program uses the Tool Command Language/ToolKit (Tcl/Tk) script language for the GUI interface. The source code of the Test Application is in the *appl/sipServer/ProxyTestApp* directory. For more information on the Test Application, see the “Test Application” chapter in the *RADVISION SIP Server Platform Programmer Guide*.

- **SimpleRedirectServer**—sample code designed to show how to use the RADVISION SIP Server Platform to build a SimpleRedirectServer. The source code of the SimpleRedirectServer is in the *samples/sipServer/SimpleRedirectServer* directory.
- **SimpleRegistrar**—sample code designed to show how to use the RADVISION SIP Server Platform to build a simple Registrar. The SimpleRegistrar implements usage of advanced features such as authenticating, and maintaining a location data base. The source code of the SimpleRegistrar is in the *samples/sipServer/SimpleRegistrar* directory.
- **SimpleStatefulProxy**—sample code designed to show how to use the RADVISION SIP Server Platform in order to build a simple stateful Proxy server. The SimpleStatefulProxy implements usage of advanced features such as maintaining a location data base. The SimpleStatefulProxy implements both the Registrar and the Proxy Server logical functions. The source code of the SimpleStatefulProxy is in the *samples/sipServer/SimpleStatefulProxy* directory.
- **SimpleStatelessProxy**—sample code designed to show how to use the RADVISION SIP Server Platform to build a simple stateless Proxy server. The SimpleStatelessProxy implements stateless Proxy Server logical function, by forwarding an incoming request to its destination according to the address resolution. The source code of the SimpleStatelessProxy is in the *samples/sipServer/SimpleStatelessProxy* directory.
- **SimpleGeneralTransaction**—sample code designed to show how to use the RADVISION SIP Server Platform for sending and receiving general transactions out-of-dialog. After the server has been started, it will automatically send a general transaction from type "METHOD". This transaction should be answered by the User Agent. The server will wait for incoming requests on port 5060, accepting valid general requests, and respond with a provisional (180) response followed immediately by an OK (200) response. The source code of the simpleGeneralTransaction is in the *samples/sipServer/SimpleGeneralTransaction* directory.
- **SimpleTlsServer**—a basic implementation of a stateful Proxy that supports TLS, which was built using the SIP Server Platform. The proxy enables registering to a database and then forwarding incoming requests. This sample also demonstrates how to:
    - Initiate TLS engines, which are responsible for TLS connection management.
    - Initiate a TLS handshake on a connected connection.
    - Use the “sips:” scheme to conduct a secure SIP session.

  The source code of SimpleTlsServer is in the *samples/sipServer/SimpleTlsServer* directory.
- **AdvancedTlsServer**—a proxy that support TLS, which was built using the SIP Server Platform. The proxy enables registering to a database and then forwarding incoming requests. This sample also demonstrates how to:

- Initiate TLS engines, which are responsible for TLS connection management.
- Initiate a TLS handshake on a connected connection.
- Use the “sips:” scheme to conduct a secure SIP session.
- Use a user-made verification callback and inspect the certificate.
- Override the SIP Stack decision on post connection assertions.

The source code of the AdvancedTlsServer is in the *samples/sipServer/AdvancedTlsServer* directory.

- **SimplePresenceServer**—sample code designed to show how to use the RADVISION Server Platform to build a simple Presence Server. The simple Presence Server implements the acceptance of a SUBSCRIBE request with event presence, and the sending of a NOTIFY (active) request from the Server to the Watcher after the SUBSCRIBE is accepted. The source code of the SimplePresenceServer is in the *samples/sipServer/SimplePresenceServer* directory. This sample code is supplied only in the package with the Events add-on.
- **SimpleWinfo**—sample code designed to show how to use the RADVISION Server Platform to build a simple Winfo application and an Events server. The SimpleWinfo accepts valid Winfo SUBSCRIBE requests and rejects invalid winfo SUBSCRIBE requests, while sending full Wnfo documents when sending a Winfo Notify. The SimpleWinfo also implements an Events server that demonstrates the handling of incoming SUBSCRIBE requests with any kind of Event header above the Events layer. If the subscription that was created matches an active Winfo subscription, a Notify(active) message will be sent to the Watcherinfo Watcher describing the status of this subscription. The source code of the SimpleWinfoServer is in the *samples/sipServer/SimpleWinfoServer* directory. This sample code is supplied only in the package with the Events add-on.
- **SimplePA**—sample code designed to show how to use the RADVISION SIP Server Platform to build a simple Presence Agent application. The simple Presence Agent implements accepting subscriptions with *event* package presence, accepting publications with event package presence, and updating Watchers by sending notification with information regarding the presentity status. The source code of the SimplePA is in the s*amples/sipServer/SimplePA* directory.
- **SimpleLdap**—sample code designed to show how to use the RADVISIONSIP Server Platform to work with the LDAP module. This sample code demonstrates a simple registrar that uses the LDAP module for a location service. The source code of the SimpleLdap is in the *samples/sipServer/SimpleLdap* directory.
- **SimpleB2BAF**—sample code designed to show how to use the RADVISION SIP Server Platform to build a simple B2BUA application. This sample code demonstrates how to easily connect a call using the Connect Transparent service and how to use the B2BAF to write your own service as shown in the General Transparent service. The source code of the simpleB2BAF is in the *samples/sipServer/SimpleB2BAF* directory. This sample code is supplied only in the package with the B2BAF Add-on Module.

- **SimpleAccounting**—sample code designed to show how to use the RADVISION SIP Server Platform to build a simple stateful Proxy server with accounting capabilities. The SimpleAccounting implements the usage of generating accounting records for various events. The SimpleStatefulProxy implements both the Registrar and the Proxy Server logical functions, as well as functions that control the accounting record creation and modification. The source code of the SimpleAccounting is in the *samples/sipServer/SimpleAccounting* directory.

Back to Top

## 5. Changes from Version 3.0

The following are non-backward-compatible changes from the SIP Server Platform version 3.0:

- The Windows development environment supports only the Debug and Release configuration. DebugIpv6DnsTls, ReleaseDNS and ReleaseTLS are no longer supported. To compile the SIP Server with DNS support, for example, please refer to the *RADVISION SIP Server Platform version 3.5 BETA Readme File*.
- The Common Core is packed as a separate package and is no longer packed with the Basic package. However, the package will be supplied along with the Basic package.
- The RvSipServerB2BAFServiceAutoRespondEventHandler() eEventType parameter was changed from RvSipServerB2BAFServiceEvent to RvInt.
- The RvSipServerB2BAFServiceConnectTranspEventHandler() eEventType parameter was changed from RvSipServerB2BAFServiceEvent to RvInt.
- The RvSipServerB2BAFServiceDisconnectTranspEventHandler() eEventType parameter was changed from RvSipServerB2BAFServiceEvent to RvInt.
- The RvSipServerB2BAFServiceGeneralTranspEventHandler() eEventType parameter was changed from RvSipServerB2BAFServiceEvent to RvInt.
- The RvSipServerB2BAFServiceConnectTranspInitCfg() hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceAutoRespondInitCfg () hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceDisconnectTranspInitCfg () hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceGeneralTranspInitCfg() hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceConnectTranspCfg hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceAutoRespondCfg hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceDisconnectTranspCfg hServiceLibMgr parameter was removed.
- The RvSipServerB2BAFServiceGeneralTranspCfg hServiceLibMgr parameter was removed.

- The B2BAF RvSipServerB2BAFServiceConstructEv() interface was changed to receive RvSipServerB2BAFServiceConstructCtxHandle as an input parameter instead of RvSipServerB2BAFServiceCtxHandle.
- The B2BAF RvSipServerB2BAFServiceInterface allocIntf functionality was enhanced to support service configuration allocation.
- The B2BAF RvSipServerB2BAFServiceInterface was enhanced to support RvSipServerB2BAFServiceInitEv, RvSipServerB2BAFServiceStoreCfgEv, RvSipServerB2BAFServiceRestoreCfgEv and RvSipServerB2BAFServiceConstructCtxHandle. For more information, please refer to the *B2BAF Programmer and Reference Guide*.
- The B2BAF RvSipServerB2BAFServiceFramework was enhanced to support sizeOfStruct, and the RvSipServerB2BAFServiceCtxHandle was replaced with RvSipServerB2BAFServiceCfgHandle. For more information, please refer to the *B2BAF Programmer and Reference Guide*.
- The RvSipServerB2BAFServiceBaseConstruct() hCtx type was changed to comply with RvSipServerB2BAFServiceConstructEv().
- The RvSipServerB2BAFServiceAutoRespondConstruct() hCtx type was changed to comply with RvSipServerB2BAFServiceConstructEv().
- The RvSipServerB2BAFServiceConnectTranspCreate() hCtx type was changed to comply with RvSipServerB2BAFServiceConstructEv().
- The RvSipServerB2BAFServiceGeneralTranspConstruct() hCtx type was changed to comply with RvSipServerB2BAFServiceConstructEv().
- The RvSipServerB2BAFServiceDisconnectTranspConstruct() hCtx type was changed to comply with RvSipServerB2BAFServiceConstructEv().
- RVSIPSERVER_B2BAF_SERVICE_DISCONNECT_TRANSP_EVENT_INVALID_REQUEST paramB was changed to support the suggested response code.
- The B2BAF no longer supports the following events:
  - RVSIPSERVER_B2BAF_TERM_EVENT_ST_REFRESH_ALER
  - RVSIPSERVER_B2BAF_TERM_EVENT_ST_NOTIFICATION
  - RVSIPSERVER_B2BAF_TERM_EVENT_ST_NEGOTIATION_FAULT

  The above events where replaced by a single event: RVSIPSERVER_B2BAF_TERM_EVENT_ST.
- The RvProxyTestApp uses Tcl/Tk v8.4 by default, but Tcl/Tk v8.3 is still supported.

Back to Top

## 6. Documentation Information

The SIP Server Platform includes the following documentation:

- *RADVISION SIP Server Platform Programmer Guide* (PDF format)
- *RADVISION SIP Server Platform Reference Guide* (PDF format)
- *RADVISION SIP Server Platform Message Layer* (PDF format)
- *RADVISION SIP Server Platform IMS Message Layer* (PDF format)
- *RADVISION Advanced Back-to-Back Application Framework Add-on Module Programmer Guide* (PDF format) [optional]
- *RADVISION Advanced Back-to-Back Application Framework Add-on Module Reference Guide* (PDF format) [optional]
- *RADVISION High Availability Add-on Module Programmer and Reference Guide* (PDF format) [optional]
- *RADVISION Events Server Add-on Module Programmer Guide* (PDF format) [optional]
- *RADVISION Events Server Add-on Module Reference Guide* (PDF format) [optional]
- *RADVISION LDAP Add-on Module Programmer and Reference Guide* (PDF format) [optional]
- *RADVISION SIP Server Platform Release Notes* (PDF format)
- *RADVISION SIP Server Platform Readme Files* (HTML format)
- *RADVISION SIP Server Platform OS Specific Readme File* (HTML format)
- *RADVISION SDP Programmer and Reference Guide* (PDF format)
- *RADVISION Porting Guide* (PDF format)

### Documentation Notes

- The Reference Guide in PDF format has been divided into the *RADVISION SIP Server Platform Reference Guide* and *RADVISION SIP Server Platform Message Layer Reference Guide* for easier handling.
- All references to Events, LDAP, SDP, Accounting, and Advanced B2B Application Framework in this manual are relevant only to the optional SIP Server Advanced Back-to-Back Application Framework Add-on Module, the SIP Server Events Add-on Module, the LDAP Add-on Module, the SDP Add-on Module, and the Accounting Add-on Module.

Back to Top

## 7. Known Issues

The list below describes the known issues in the current release. A customer support issue identification ID is given in braces where applicable.

1. When working with the LDAP Add-on Module, using OpenLDAP version 2.1.25 on Windows, in a multithreaded environment, runtime problems may occur. The next OpenLdap versions may fix this issue.
2. When working on the Linux OS, the application cannot open IPv4 and IPv6 addresses using the same port. This is a restriction of the OS.
3. Messages larger than the value specified for sendReceivedBufferSize in the configuration (2048 bytes by default) will be discarded.
4. The maximal Expires header value (delta seconds) is limited to 2^31-1 rather than 2^32-1, as specified in RFC 3261.
5. Server-side DNS is supported only for the first entry in the DNS resolution.
6. The Authentication and Authorization headers allow setting only one parameter in the other param field.
7. ACKs cannot be resent on failure.
8. The Stack adds the Proxy-Authorization header to outgoing CANCEL requests even though it is not required according to RFC 3261. The application can choose to remove this header in the message to send the callback.
9. The Stack parser fails to parse a Retry-After header with an empty comment. For example, the following header will fail to be parsed: Retry-After: 120 ()

   The application can register to the “bad syntax” callback and fix this problem at runtime.
10. A SIP message can include header field rows that consist of a header field name and zero or more header field values separated by commas. In such a case, if one of the header values contains a syntax error, the message is not parsed correctly. The correct header values are duplicated in the message object.
11. When working with heavy multithreaded TCP stress on the server side, Windows may report a faulty FD_CLOSE event on newly opened server connections. The Faulty FD_CLOSE event may be reported even if no FIN or RST was sent on the network. If such a faulty event is reported by Windows, the SIP Server will close the server connection. This may result in a failure to accept a call.
12. The SIP Server does not allow listening for incoming requests on port numbers 0 and 65535.
13. The SIP Server does not allow listening on local loop addresses (127.0.0.1).
14. Unexpected behavior may occur upon SIP Server startup if the syntax of a configuration parameters is invalid (00046906).
15. When running the ProxyTestApp on Win64 bits operating systems, you may get an error when the application exits.

16. When compiling the SIP Server with a log level different from RV_LOGLEVEL_ALL, the following compilation warnings may appear:
    - Un-referenced local variable
    - Local variable is initialized but not referenced
    - Un-referenced local function has been removed
17. When loading a workspace file (*.sln/*.dsw) into Visual Studio 2005/Visual Studio 6, there may be missing projects files (*.vcproj/*.dsp) because you did not acquire the SIP Server add-on features. You should remove all the missing projects and save to workspace to avoid this warning the next time you are loading the project.
18. When compiling the SIP Server with a RV_THREADNESS_TYPE = RV_THREADNESS_SINGLE, the following compilation warnings may appear:
    - Un-referenced local variable
    - Local variable is initialized but not referenced
    - Un-referenced local function has been removed
19. If a socket is closed during the select/poll system call, an error might occur.
20. When the SIP Server receives a UDP message with a Content-length header value that does not indicate the actual length of the message body, the SIP Server updates the Content-length header with the correct value and continues with regular message processing.
21. A comma (,) is not allowed in the username part of a SIP address that is not inside <> due to syntax ambiguity in RFC 3261.
22. The SIP Server needs to be compiled for VxWorks with the -DINET6 compilation flag if the OS target image was built with IPv6 support. This flag can be added (if needed) in the *common/examples/vxworks.diab.default.mak* file or the *common/examples/vxworks.gnu.default.mak* file, through the CFLAGS variable.
23. If TLS Renegotiation is used on Client and Server sides of the connection, and both agents use OpenSSL, the situation may lead to a communication deadlock due to OpenSSL limitations.
24. The RvProxyTestApp does not support the RV_SELECT_POLL nor the RV_SELECT_DEVPOLL compilation flags.
25. The GNU C 4.2 Compiler version is not supported.

Back to Top

## 8. Contacting RADVISION Support

The SIP Server Platform package supplied herewith is a Release version. If you are using this version, please forward problem reports directly to RADVISION support. We are also interested in any comments made on the contents of this package, the functionality supplied by the SIP Server Platform API, the documentation files and any other issues regarding this delivery. Your input is valuable to us and allows our staff to deliver future upgrade versions that best suit your needs.

| | |
|---|---|
| International Headquarters:<br>RADVISION Ltd.<br>Tel: (+972) 3 767 9555<br>Fax: (+972) 3 767 9550<br>SWsupport@radvision.com | USA Headquarters:<br>RADVISION, Inc.<br>Tel: (+1) 201 689 6380<br>Fax: (+1) 201 689 6301<br>ToolKitSupport@radvision.com |
| EMEA Headquarters<br>RADVISION UK<br>Tel: (+44) 208 757 8826<br>Fax: (+44) 208 757 8818<br>EMEA_support@radvision.com | APAC Headquarters<br>RADVISION Hong Kong<br>Tel: (+852) 2 8014 070<br>Fax: (+852) 2 8014 071<br>APACsupport@radvision.com |

**Note** When you contact RADVISION technical support, please write the following in the **Subject** field of your e-mail: "SIP Server Platform version 3.5 question" and a summary/headline describing the problem.

RADVISION web site http://www.radvision.com

Back to Top