Pulsesecure
Steel-Belted Radius

RSA SecurID Ready Implementation Guide

Last Modified: January 15, 2016

Partner Information

Product Information	
Partner Name	Pulsesecure/Pulsesecure Networks
Web Site	www.Pulsesecure.net
Product Name	Steel-Belted Radius
Version & Platform	6.1.7 on Windows, Linux, and Solaris
Product Description	Steel-Belted Radius is a complete implementation of the widely used IETF standards-track RADIUS (Remote Authentication Dial-In User Service) protocols. It acts as a security gateway to your LAN that authenticates, authorizes and accounts for all remote and wireless LAN access. It interfaces with a wide variety of network access servers, including Wireless Access Points, VPN and Dial-in servers and easily authenticates remote and WLAN users against your existing security infrastructure.

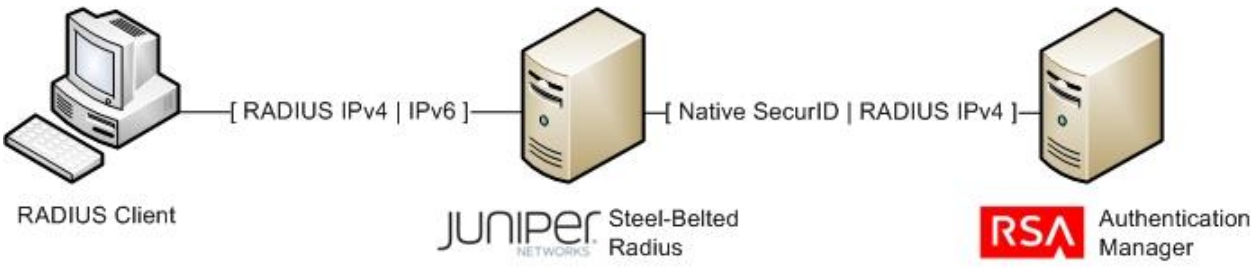
Solution Summary

Steel-Belted Radius interfaces with a wide variety of network access equipment, and authenticates remote and wireless LAN (WLAN) users against numerous back-end databases. This allows for consolidation of administration for all remote and WLAN users.

Steel-Belted Radius can be configured to communicate with RSA Authentication Manager via RSA's native SecurID protocol. This integration allows RSA's two factor authentication to be used when authenticating users to network resources.

Steel-Belted Radius employs a dual IP stack to support both IPv4 and IPv6 address types. This feature enables RSA Authentication Manager to provide authentication services to hosts on IPv6 networks.

RSA Authentication Manager supported features Steel-Belted Radius 6.1.7	
RSA SecurID Authentication via Native RSA SecurID Protocol	Yes
RSA SecurID Authentication via RADIUS Protocol	No
On-Demand Authentication via Native SecurID Protocol	Yes
On-Demand Authentication via RADIUS Protocol	No
Risk-Based Authentication	No
Risk-Based Authentication with Single Sign-On	No
RSA Authentication Manager Replica Support	Yes
Secondary RADIUS Server Support	No
RSA SecurID Software Token Automation	No
RSA SecurID SD800 Token Automation	No
RSA SecurID Protection of Administrative Interface	No



Note: Pulsesecure Networks SRB is now part of Pulsesecure

Authentication Agent Configuration

Authentication Agents are records in the RSA Authentication Manager database that contain information about the systems for which RSA SecurID authentication is provided. All RSA SecurID-enabled systems require corresponding Authentication Agents. Authentication Agents are managed using the RSA Security Console.

The following information is required to create an Authentication Agent:

- Hostname
- IP Addresses for network interfaces

Set the Agent Type to "Standard Agent" when adding the Authentication Agent. This setting is used by the RSA Authentication Manager to determine how communication with Steel-Belted Radius will occur.

Note: Hostnames within the RSA Authentication Manager / RSA SecurID Appliance must resolve to valid IP addresses on the local network.

Please refer to the appropriate RSA documentation for additional information about creating, modifying and managing Authentication Agents.

RSA SecurID files

RSA SecurID Authentication Files	
Files	**Location**
sdconf.rec	%SystemRoot%\System32 or $VAR_ACE
Node Secret	%SystemRoot%\System32 or $VAR_ACE
sdstatus.12	%SystemRoot%\System32 or $VAR_ACE
sdopts.rec	%SystemRoot%\System32 or $VAR_ACE

Note: The appendix of this document contains more detailed information regarding these files.

Important: This version of Steel-Belted Radius uses a new version of the RSA Authentication libraries that changes the encryption format of the node secret file.

If you are upgrading from a previous version of Steel-Belted Radius, you must clear the node secret or convert it using a tool available from RSA. Instructions for clearing the node secret can be found in the Appendix section of this document.

Note: $VAR_ACE is an environment variable defined by the Steel-Belted Radius daemon's init script on Solaris and Linux platforms. Refer to this script on your platform for the location of the SecurID files, and where to copy the sdconf.rec configuration file.

If $VAR_ACE is not defined, the system will read and write configurations in /var/ace on the system.

Partner Product Configuration

Before You Begin

This section provides instructions for configuring Steel-Belted Radius with RSA SecurID Authentication. This document is not intended to suggest optimum installations or configurations.

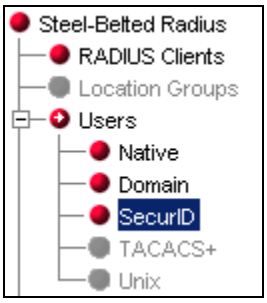
It is assumed that the reader has both working knowledge of all products involved, and the ability to perform the tasks outlined in this section. Administrators should have access to the product documentation for all products in order to install the required components.

All Steel-Belted Radius components must be installed and working prior to the integration. Perform the necessary tests to confirm that this is true before proceeding.

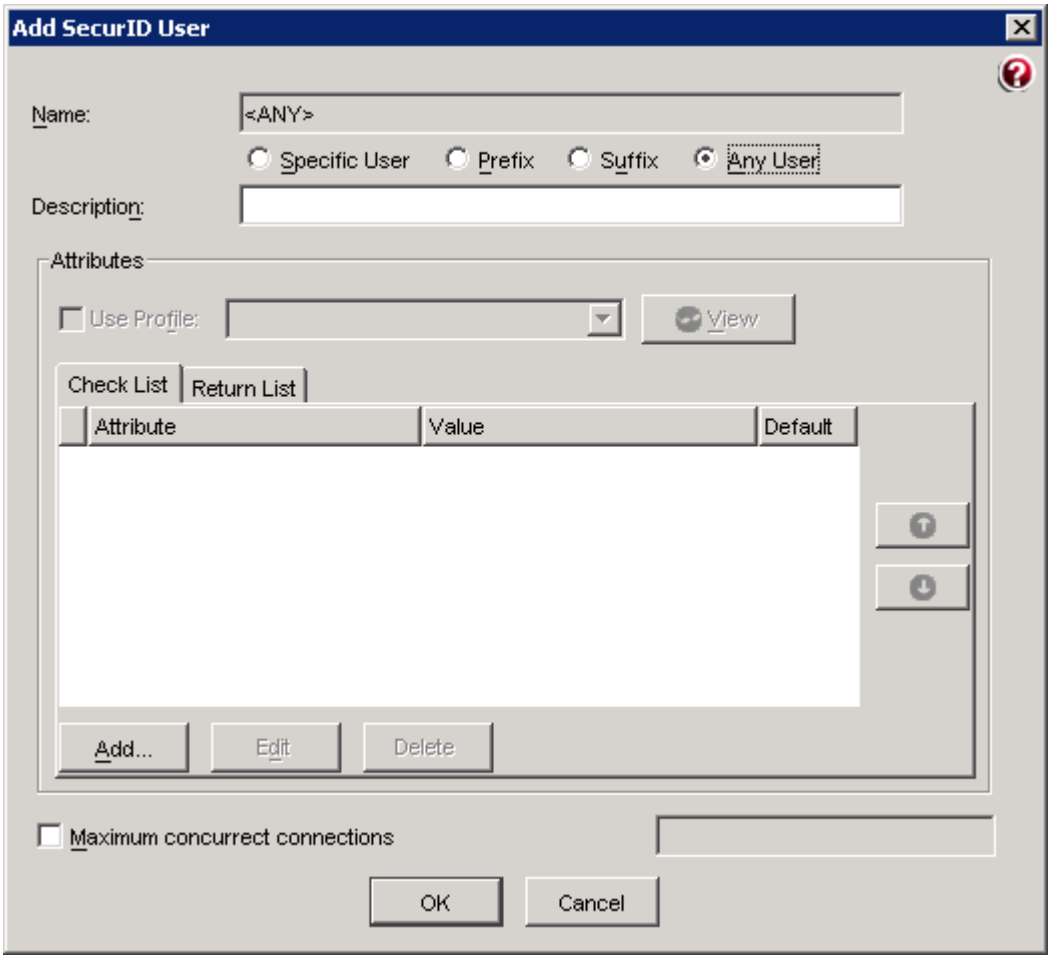
Configuring Steel-Belted Radius for SecurID Authentication

Steel-Belted Radius supports RSA SecurID authentication using RSA's native SecurID protocol. The following instructions will enable SecurID authentication on a Steel-Belted Radius server running in a Windows environment.

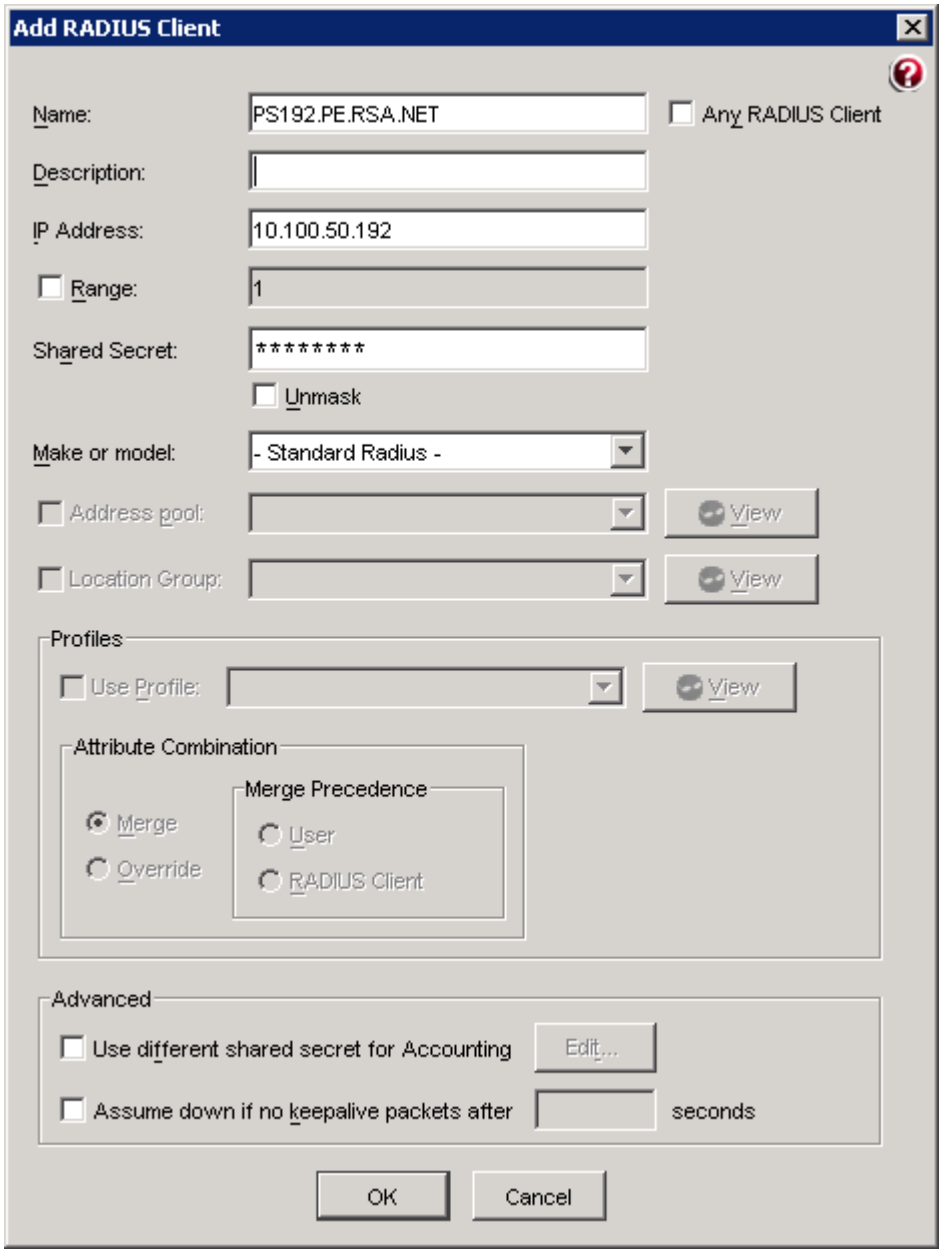
1. Copy the **sdconf.rec** file obtained from the RSA Security Console to **%SystemRoot%\System32** or the **SysWOW64** on your SBR server. If the Steel-Belted Radius service is running, restart it.
2. Using the **SBR Administrator** utility, expand the **Users** node. The **SecurID** node should be available



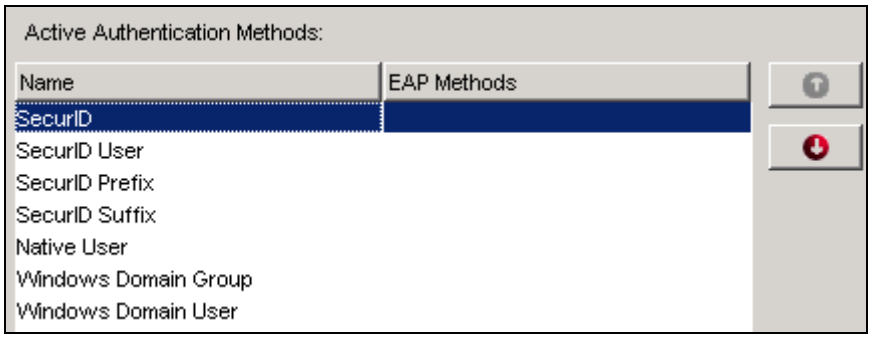
3. Click on the **SecurID** node and click **Add** to create a new user. Steel-Belted Radius can be configured to authenticate SecurID users that match a username, all usernames that match a certain prefix or suffix, or all users, as determined by the authentication method order (covered later in this document). Select a user type and provide a **Name** if you selected a type other than **Any User**. Click **OK** to create the new user.



4. Click on the **RADIUS clients** node and click **Add** to create a new RADIUS client. Specify the hostname and IP address of your RADIUS client, as well as the shared secret. Specify the make or model as **Standard Radius**. Click **OK** to create the RADIUS client.



5. Click on the **Authentication Policies > Order of Methods** node. A number of active authentication methods are displayed in the right column. This is the order in which the RADIUS server will attempt to match incoming user authentication requests, and must reflect how you chose to configure SecurID users. Since this example uses the <ANY> SecurID user, the **SecurID** method is at the top of the list. You may also wish to remove authentication methods that are not necessary from this list.



6. Make the following changes to the **[SecurID]** section of **radius.ini** to allow SBR to cache SecurID passcodes. This must be enabled if you have ISDN users because it makes it possible for those users to open a second B-channel during authentication. The radius.ini file is found in the **service** subdirectory of your SBR installation folder.

```
[SecurID]
CachePasscodes          = yes
SecondsToCachePasscodes = 60
```

7. System-generated PINs are not enabled by default. To enable the RADIUS server to deliver system-generated SecurID PINs to users, make the following changes to the **[Configuration]** section of **securid.ini**. This file is found in the **service** subdirectory of you SBR installation folder.

```
[Configuration]
Enable = 1
CheckUserAllowedByClient = 0
DefaultProfile = DEFAULT
AllowSystemPins = 1
```

8. If you plan to use RSA SecurID authentication with EAP Generic-Token protocol support, edit the **[SecurID]** section of **eap.ini**, located in the **Service** subfolder of your SBR installation folder. Verify that the EAP settings in this section are enabled (remove the semicolon from the start of each line) and configured properly according to your requirements.
9. Restart the Steel-Belted Radius service. The server is now configured for SecurID authentication foe a created radius client.

Certification Checklist for RSA Authentication Manager

Date Tested: May 3, 2013

Certification Environment		
Product Name	**Version Information**	**Operating System**
RSA Authentication Manager	8.0	Virtual Appliance
Pulsesecure Steel-Belted Radius	6.1.7	Windows Server 2008 R2 CentOS 4

Mandatory Functionality			
RSA Native Protocol		**RADIUS Protocol**	
New PIN Mode			
Force Authentication After New PIN	✓	Force Authentication After New PIN	N/A
System Generated PIN	✓	System Generated PIN	N/A
User Defined (4-8 Alphanumeric)	✓	User Defined (4-8 Alphanumeric)	N/A
User Defined (5-7 Numeric)	✓	User Defined (5-7 Numeric)	N/A
Deny 4 and 8 Digit PIN	✓	Deny 4 and 8 Digit PIN	N/A
Deny Alphanumeric PIN	✓	Deny Alphanumeric PIN	N/A
Deny PIN Reuse	✓	Deny PIN Reuse	N/A
Passcode			
16-Digit Passcode	✓	16-Digit Passcode	N/A
4-Digit Fixed Passcode	✓	4-Digit Fixed Passcode	N/A
Next Tokencode Mode			
Next Tokencode Mode	✓	Next Tokencode Mode	N/A
On-Demand Authentication			
On-Demand Authentication	✓	On-Demand Authentication	N/A
On-Demand New PIN	✓	On-Demand New PIN	N/A
Load Balancing / Reliability Testing			
Failover (3-10 Replicas)	✓	Failover	N/A
No RSA Authentication Manager	✓	No RSA Authentication Manager	N/A

MRQ

✓ = Pass ✗ = Fail N/A = Not Applicable to Integration

Note: Complete testing was performed using clients on both IPv4 and IPv6 networks.

Appendix

Partner Integration Details	
RSA SecurID API	8.1.1
RSA Authentication Agent Type	Standard Agent
RSA SecurID User Specification	Designated Users
Display RSA Server Info	No
Perform Test Authentication	No
Agent Tracing	Yes

API Details:

This version of Steel-Belted Radius uses a new version of the RSA Authentication libraries that changes the encryption format of the node secret file.

If you are upgrading from a previous version of Steel-Belted Radius, you must clear the node secret or convert it using a tool available from RSA. Instructions for clearing the node secret can be found below. To convert the node secret, you must use the **agent_nsload** utility found in the RSA Authentication Agent SDK 8.1 SP1. This tool is documented in the API developer's guide included with the SDK.

Node Secret:

The node secret is stored in **%SystemRoot%\System32** or **$VAR_ACE** (Linux/Solaris) on the SBR server. To clear the node secret from the server, delete the **securid** file from this directory. Be sure to clear the node secret on your Authentication Manager instance as well, or authentication will fail.

sdconf.rec:

The sdconf.rec file is stored in **%SystemRoot%\System32** or **$VAR_ACE** (Linux/Solaris) on the SBR server. Certain changes to Authentication Manager, such as changing IP addresses or adding and removing replica servers, require refreshing this file with a current copy. If you must replace this file, obtain a fresh copy using the RSA Security Console and copy it to this location, overwriting any previous version.

Agent Tracing (Windows):

Using Regedit, locate the HKEY_LOCAL_MACHINE\Software\SDTI\ACECLIENT key and create 2 DWORD values: **tracelevel** and **tracedest.**

The value **tracelevel** specifies the verbosity and the categories of messages produced by the code. The value **tracedest** controls the output destination of the trace messages.

tracedest VALUES:

```
SDITRACE_EVENT_LOG  0x00000001   // messages to event log
SDITRACE_CONSOLE    0x00000002   // messages to console
SDITRACE_LOGFILE    0x00000004   // messages to logfile (aceclient.log)
SDITRACE_DEBUGGER   0x00000008   // messages to debugger output
SDITRACE_NOFILELINE 0x80000000   // no file and line information
```

The SDITRACE_NOFILELINE value can be combined with any of the other values to stop the display of file and line number information. The logfile is **%SystemRoot%\ACECLIENT.LOG** but can be changed by creating a **REG_SZ:tracefile** value and specifying the file pathname.

tracelevel VALUES:

```
SDITRACEING_OFF     0x000000000  // All messages off
SDITRACEING_ON      0x000000001  // All messages marked with this level on
SDITRACEING_ENTRY   0x000000002  // All entrypoints use this
SDITRACEING_EXIT    0x000000004  // All function returns use this
SDITRACEING_FLOW    0x000000008  // All logic flow control use this (ifs)
SDITRACEING_GRP1    0x000000010  // Old SDITRACE macros use this (see dbglib.h)
```

The hex value 0xF gives the complete set of tracing. The values can be combined to produce multiple sets of trace messages.

> **Note: Using the SDITRACE_CONSOLE value can cause the service applications to access violate during logoff. Use only for real time debugging situations.**

Agent Tracing (Linux / Solaris):

Agent tracing in Solaris and Linux is similar to Windows, except that instead of editing the registry, you must configure the Steel-Belted Radius daemon's initialization script to export environment variables that configure logging.

Configure a variable called **RSATRACELEVEL** and give it a decimal value to configure the tracing level. These values are identical to the values for Windows, with a value of 0 disabling tracing and a value of 15 enabling all trace options.

Configure a variable called **RSATRACEDEST** and give it a path to the desired output log file.

IPv6 support:

Steel-Belted Radius employs a dual IP stack to support both IPv4 and IPv6 address types. This feature enables the Steel-Belted Radius to proxy authentication requests from hosts on IPv6 networks to RSA Authentication Manager servers on IPv4 networks.