Network RADIUS
FreeRADIUS

RSA SecurID Ready Implementation Guide

Last Modified: December 7, 2011

Partner Information

Product Information	
Partner Name	Network RADIUS SARL
Web Site	www.networkradius.com, www.freeradius.org
Product Name	FreeRADIUS
Version & Platform	2.12 for Linux
Product Description	FreeRADIUS is an open-source RADIUS server that provides robust and highly configurable RADIUS authentication services. The server is capable of simultaneously running on both IPv4 and IPv6 interfaces and is capable of sending as an authentication proxy for other RADIUS servers. The FreeRADIUS project is sponsored by Network RADIUS SARL, which also offers enterprise support packages for the product.
Support URL	http://networkradius.com/support

Network RADIUS SARL

Solution Summary

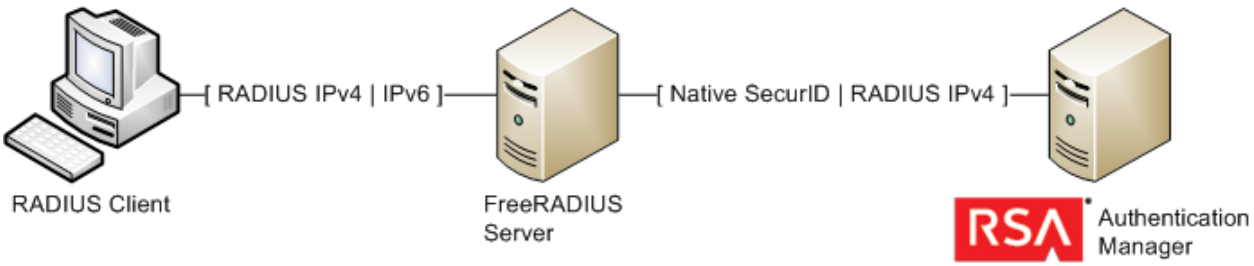
FreeRADIUS provides support for SecurID two-factor authentication with an authentication module. Once compiled with the RSA Authentication Agent libraries (available to RSA Authentication Manager customers), this module allows FreeRADIUS to provide two-factor token-based authentication over the native SecurID protocol.

FreeRADIUS is also able to proxy RADIUS requests to the RSA RADIUS server. When configured this way, incoming RADIUS requests that meet certain criteria are sent as proxy requests to the RADIUS server built in to RSA Authentication Manager.

FreeRADIUS 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.

Note: For support on this solution, contact Network RADIUS (http://networkradius.com/support).

RSA SecurID supported features	
FreeRADIUS 2.12	
RSA SecurID Authentication via Native RSA SecurID Protocol	Yes
RSA SecurID Authentication via RADIUS Protocol	Yes
On-Demand Authentication via Native SecurID Protocol	Yes
On-Demand Authentication via RADIUS Protocol	Yes
On-Demand Authentication via API	No
RSA Authentication Manager Replica Support	Yes
Secondary RADIUS Server Support	Yes
RSA SecurID Software Token Automation	No
RSA SecurID SD800 Token Automation	No
RSA SecurID Protection of Administrative Interface	No



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 FreeRADIUS will occur.

A RADIUS client that corresponds to the Authentication Agent must be created in the RSA Authentication Manager in order for FreeRADIUS to communicate with RSA Authentication Manager. RADIUS clients are managed using the RSA Security Console.

The following information is required to create a RADIUS client:

- Hostname
- IP Addresses for network interfaces
- RADIUS Secret

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 and RADIUS clients.

RSA SecurID files

RSA SecurID Authentication Files	
Files	**Location**
sdconf.rec	/var/ace
Node Secret	/var/ace
sdstatus.12	/var/ace
sdopts.rec	/var/ace

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

Partner Product Configuration

Before You Begin

This section provides instructions for configuring the FreeRADIUS 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 FreeRADIUS components must be installed and working prior to the integration. Perform the necessary tests to confirm that this is true before proceeding.

Overview

FreeRADIUS can communicate with RSA Authentication Manager either by proxying authentication requests to Authentication Manager's built-in RADIUS server, or by sending authentication requests over the native SecurID protocol. The Native SecurID protocol has the advantage that failover and load balancing are handled automatically. When proxying requests over the RADIUS protocol, failover and load-balancing between Authentication Manager servers is supported, but this must be configured in the FreeRADIUS site's configuration files.

This guide will describe both configurations. Choose the one that meets your organization's needs.

Configuring SecurID Authentication – Native SecurID Protocol

In order to configure FreeRADIUS to communicate over the native SecurID protocol, you must compile and install the **rlm_securid** authentication module available from the FreeRADIUS project.

> **Note: Compiling the rlm_securid module requires you to obtain the RSA Authentication Agent client libraries from the RSA Authentication Agent SDK, available from RSA SecurCare Online at https://knowledge.rsasecurity.com**
>
> **Compiling and installing the rlm_securid module is outside of the scope of this document. For instructions, please visit the homepage of the FreeRADIUS project.**

1. Once the rlm_securid module is successfully, built and installed, define an Auth-Type for SecurID to the sites for which you want to enable SecurID Authentication. Sites are defined in **/etc/freeradius/sites-available/** directory. Add a SecurID Auth-Type to the file(s) that correspond to the site(s) that you want to enable for SecurID authentication. For example, in order to add SecurID authentication to the DEFAULT site, add the SecurID Auth-Type to the **authenticate** section of the **default** file.

```
authenticate {
        Auth-Type SecurID {
                securid
        }
}
```

2. Once the Auth-Type is defined, be sure to set this Auth-Type for the users that you wish to enable for SecurID authentication by editing the **users** file in **/etc/freeradius/**. The Auth-Type must match what you have defined in your sites.

> **Note: There are many granular options for classifying sites and users. Consult the FreeRADIUS documentation for full details.**

```
#adds the SecurID Auth-Type to 'Securid_User_1'
Securid_User_1      Auth-Type := SecurID

#adds the SecurID Auth-Type to all users
DEFAULT Auth-Type := SecurID
```

3. **Restart** the FreeRADIUS daemon to apply these changes. Once the server restarts, it will begin enforcing two-factor authentication for the configured sites and users.

Configure SecurID Authentication – RADIUS Proxy

FreeRADIUS can be configured to proxy incoming RADIUS requests that match specified criteria to Authentication Manager's built-in RADIUS server. When configured this way FreeRADIUS acts as a RADIUS proxy to Authentication Manager.

In order to configure FreeRADIUS to send proxy RADIUS requests, you must define one or more of your Authentication Manager servers as home servers, configure failover options, and assign the failover pool to one or more realms used for authentication.

> **Note: You must have RADIUS servers configured in your Authentication Manager deployment in order to accept proxy RADIUS requests. Refer to Authentication Manager's documentation for detailed instructions on configuring RSA RADIUS.**

1. Define your FreeRADIUS server as a RADIUS client using the RSA Security Console. Be sure to note the shared secret you define, as it will be needed in the following configuration steps.
2. On the FreeRADIUS server, edit the **proxy.conf** configuration file using a text editor.

3. Add **home_server** entries for your Authentication Manager server. Each entry can have failover options defined according to the FreeRADIUS documentation. These options will vary according to your requirements.

> **Note: Be sure that the entry for the secret matches the shared secret you established in the RSA Security Console.**

```
home_server vm3143.pe.rsa.net {
        type = auth
        ipaddr = 10.100.53.143
        port = 1812
        secret  = 12345678
        require_message_authenticator = yes
        response_window = 15
        zombie_period = 40
        status_check = status-server
        check_interval = 15
        num_answers_to_alive = 3
        max_outstanding = 65536
}

home_server vm3144.pe.rsa.net {

       ...Options not shown...

}

home_server vm3145.pe.rsa.net {

       ...Options not shown...

}
```

4. Add a **home_server_pool** entry for your configured home servers. This entry will govern failover of the proxy requests if one of your configured home server's goes down. In this example, a simple fail-over style is used. The proxy-requests are sent to the home_servers in the order they are listed, failing over to backup servers if the primary server is unresponsive.

> **Note: There are a number of load-balancing and failover options available. Refer to the FreeRADIUS documentation for full details on failover and load-balancing options.**

```
home_server_pool authman_failover {
        type = fail-over
        home_server = vm3143.pe.rsa.net
        home_server = vm3144.pe.rsa.net
        home_server = vm3145.pe.rsa.net
}
```

5. Finally, define a **realm** that will use the home server pool. You may wish to define a special realm for SecurID requests, add the home server pool to an existing realm, or simply use the DEFAULT realm, as is shown in this example.

```
realm DEFAULT {
        auth_pool = authman_failover
}
```

6. **Save** the configuration file and **restart** FreeRADIUS. Incoming authentication requests that match your defined realm settings will now be proxied to Authentication Manager for validation.

Certification Checklist for RSA Authentication Manager

Date Tested: December 7, 2011

Certification Environment		
Product Name	**Version Information**	**Operating System**
RSA Authentication Manager	7.1 SP4	Windows Server 2003
FreeRADIUS / rlm_securid module	2.12	Ubuntu Server 11.04 64-bit
RSA Authentication Libraries	8.1 SP1	Ubuntu Server 11.04 64-bit

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

MRQ

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

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

Known Issues

Versions of FreeRADIUS prior to 2.12 had an issue with proxying RADIUS requests where the RADIUS server daemon would terminate unexpectedly if none of the home servers were alive. This issue has been verified as fixed in version 2.12 of the server.

Appendix

Partner Integration Details	
RSA SecurID API	API Version 8.1 SP1 (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:

Because it is up to the user to supply the API shared libraries, it is possible that other versions may be used. RSA recommends obtaining the latest version of the Authentication Agent SDK, which, at the time of this certification, is version 8.1 SP1 (8.1.1).

Node Secret:

The node secret is stored in **/var/ace** in a file called **securid**. To clear the node secret on the server, delete this file.

sdconf.rec:

The sdconf.rec configuration file is stored in **/var/ace**. To update this configuration, replace this file with a fresh copy from your Authentication Manager deployment.

sdopts.rec:

The optional sdopts.rec configuration can be placed in **/var/ace**. This file can contain certain authentication agent overrides that may be useful in your configuration. Refer to the RSA Authentication Agent SDK's Developer Guide for more information on this configuration file.

sdstatus.12:

The sdstatus.12 configuration file contains cached information on the RSA Authentication Manager server topology, and is stored in **/var/ace**.

Agent Tracing:

Agent tracing can be enabled by exporting environment variables. These variables can be exported as part of the FreeRADIUS daemon's init script, or you can export them in your current session and start FreeRADIUS manually. The FreeRADIUS init script is found in **/etc/init.d/freeradius**.

To enable tracing, export two variables: **RSATRACELEVEL** and **RSATRACEDEST**.

RSATRACEDEST is the location of the log file where log events will be written. This can be a relative or absolute path.

RSATRACELEVEL is the level of logging that should take place, and can have the following values. These values can be added to combine event levels. For example, setting the value to 3 enables logging for both regular messages and function entry points.

RSATRACELEVEL values	
Value	**Description**
0	Disable logging
1	Log regular messages
2	Log function entry points
4	Log function exit points
8	Log all logic flow controls