Lucent Technologies
Bell Labs Innovations

RADIUS Server 2.1 Release Note

June 23, 1999

The Lucent Remote Access RADIUS server 2.1 with support for RADIUS proxy, iPass roaming, and ActivCard is now available in binary form for the following platforms, and in source form:

- IBM RS6000 AIX 4.2
- Alpha Digital UNIX 4.0
- BSD/OS 3.0
- HP-UX 10.20
- Slackware Linux 2.0
- Redhat Linux 5.2
- SGI IRIX 6.3
- SunOS 4.1.4
- Solaris 2.5.1
- Solaris x86 2.5.1

Note – This release removes the obsolete RADPASS feature. In addition, User-Name values with embedded "at" signs (@) are now treated as proxy realms.

RADIUS server 2.1 includes a new dictionary file with support for the Class, LE-Advice-of-Charge, and LE-Terminate-Detail attributes added in ComOS® 3.8. Attributes added in ComOS 3.9 and 4.1 are also included.

The **sradiusd** daemon provides support for ActivCard on platforms supported by ActivEngine 2.1: AIX, HP-UX, SunOS, and Solaris.

All other files are the same as in RADIUS server 2.0.1. Support is available only for customers owning Lucent PortMaster products. Information on contacting Lucent Remote Access technical support is listed at the end of this release note.

The **radiusd** daemon uses GDBM 1.7.3 instead of NDBM on some systems; the source for GDBM is available for free from **ftp://ftp.gnu.org/pub/gnu/gdbm/** and other Free Software Foundation (FSF) distribution sites.

RADIUS Server Features

- Y2K Compliance
- Proxy RADIUS
- ActivCard

- iPass Support
- Accounting Signatures Now Required
- Vendor-Specific Attributes
- Virtual Ports
- Alternate Password File
- Address Binding
- Improved Messages
- Enhanced Debugging
- Bugs Fixed
- RADIUS test program radtest
- How Proxy RADIUS Works and How to Configure It
- Upgrading
- Contacting Lucent Remote Access Technical Support

RADIUS Server Features

RADIUS server 2.1 supports the following features:

Y2K Compliance

RADIUS server 2.1 is Y2K compliant. It treats all dates internally as 32-bit unsigned integers or time_t, and prints years in 4-digit format (for example, 1999). You must ensure that the operating system you are running the RADIUS server on is also Y2K compliant. All PortMaster products are Y2K compliant because they do not track the year.

Proxy RADIUS

Proxy RADIUS is a feature in which one RADIUS server can forward an authentication request to a remote RADIUS server, and return its reply to the network access server (NAS). A common use for proxy RADIUS is roaming. Roaming permits two or more ISPs to allow each other's users to dial in to either ISP's network for service.

For more information on proxy RADIUS, see “How Proxy RADIUS Works and How to Configure It“ later in this release note.

ActivCard

RADIUS server 2.1 now supports ActivCard as well as SecurID for authentication. Do the following to authenticate a user with ActivCard:

1. Install the new ActivCard server on the same host as the RADIUS server or another host.

2. Create the **/etc/raddb/config.aeg** file on your RADIUS server host describing the parameters used to connect to the ActivCard server.

3. Use "Auth-Type = ActivCard" as a check item for the user.

iPass Support

RADIUS server 2.1 now supports the iPass protocol. Do the following to use iPass:

1. Register at the iPass website **http://www.ipass.com/**.

2. Add the **ipass** keyword to the appropriate entries in your **/etc/raddb/proxy** file.

3. Run the **iradiusd** binary instead of **radiusd**.

Direct any problems with the iPass support to iPass technical support first; iPass will contact Lucent Remote Access, if necessary.

Accounting Signatures Now Required

Earlier releases of the Lucent Remote Access RADIUS server logged RADIUS accounting packets even if their request authenticators were invalid. This behavior provided backwards compatibility with ComOS 3.3 and earlier releases. RADIUS server 2.1 now discards invalid accounting packets and logs an error message.

Caution – If you have any PortMaster running ComOS 3.3 or earlier, you must upgrade it to ComOS release 3.3.1 or later to use RADIUS server 2.1.

The **-o** flag is provided for backwards compatibility with noncompliant RADIUS clients. If **radiusd** is run with the **-o** flag, it logs unsigned accounting records, and flags them with "**Request-Authenticator = None**". If **radiusd** is run without the **-o** flag, it does not log unsigned accounting records.

Vendor-Specific Attributes

RADIUS server 2.1 supports vendor-specific attributes in accounting-request packets to support the LE-Advice-of-Charge and LE-Terminate-Detail attributes added in ComOS 3.8. ComOS releases are available at **http://www.livingston.com/forms/one-click-dnload.cgi** and via FTP at **ftp://ftp.livingston.com/pub/le/upgrades/**.The dictionary file uses the following syntax to define vendor-specific attributes that follow the suggested format in RFC 2138:

```
#
# Vendor-Specific attributes use the SMI Network Management Private# nterprise
Code from the "Assigned Numbers" RFC.
#
VENDOR            Livingston       307
# Livingston Vendor-Specific Attributes (requires ComOS 3.8 and RADIUS 2.1)
ATTRIBUTE        LE-Terminate-Detail      2        string  Livingston
ATTRIBUTE        LE-Advice-of-Charge      3        string  Livingston
```

LE-Terminate-Detail is a string, included in RADIUS Accounting Stop records generated by ComOS 3.8, that contains a detailed description of the reason for session termination.

LE-Advice-of-Charge is a string containing the Advice of Charge information (if any) provided on the ISDN D channel by the telephone company.

Virtual Ports

If the file **/etc/raddb/vports** exists, it restricts the number of logins to each telephone number listed in the file. The first column of the file contains the Called-Station-Id, and the second column contains the number of logins permitted into that telephone number.

This **virtual ports** feature provides only an approximate access control. Logins occurring before radiusd starts running are not considered in the count, nor are accounting records that go to the backup accounting server.

To use this feature you must run radiusd with the **-s** (single-threaded) flag, and your must run the authentication and accounting servers on the same host.

This feature does not provide simultaneous login limits for users. It is based on Called-Station-Id, not Calling-Station-Id.

Alternate Password File

You can use the **-f** flag with **radiusd** to specify an alternative to the password file **/etc/passwd**.

Address Binding

The **-i** <Address> flag to **radiusd** instructs the RADIUS server to bind to the specified IP address to listen for requests, instead of binding to any address. This address binding is useful for running radiusd on multihomed hosts.

Improved Messages

- The Calling-Station-Id, where known, is now included in the syslog message for many kinds of rejected access-requests, to help you identify where failed login attempts are dialing from. Currently, this value is logged to syslog for unknown users and for failed "Auth-Type = System" logins. For example:

  ```
  Jul 10 21:10:50 ra radius[14870]: unix_pass: password for "bob" at 5551234
  failed
  ```

  The actual syslog message appears on one line, but is broken into two lines here for legibility.

- Other log messages are now more detailed. For example:

  ```
  Jul 10 21:10:50 ra radius[14870]: forwarding request from
  192.168.96.6/1093.139 to 172.16.3.24/1645.17 for edu.com
  ```

  The numbers after the slash are the UDP port (1645) and the RADIUS message ID (17) for easier tracking. The actual syslog message appears on one line, but is broken into two lines here for legibility.

Enhanced Debugging

Sending a SIGUSR1 signal to **radiusd** now turns on debugging, and sending SIGUSR2 turns off debugging. Either signal, or exiting **radiusd**, logs a short summary of the daemon's activity. The format is subject to change, but for this release the summary looks like the following examples. The actual syslog message appears on one line, but is broken into two lines here for legibility.

Example 1

```
Mar 19 23:10:50 ra radius[14870]: counters 5 8 / 2 4 / accept 4 reject 1
challenge 0 response 8
```

In this example, five packets were received on the RADIUS port (usually 1645 unless changed with the **-p** flag), eight packets were received on the RADIUS accounting port. Two RADIUS proxy replies were received, and four RADIUS accounting proxy replies were received. The RADIUS server sent four access-accepts, one access-reject, no access-challenges, and eight accounting responses.

Example 2

```
Jul 28 09:56:01 ra radius[19340]: memory usage = pair 8/35/4784 peer 0/0/0 req
1/4/570 buf 1/4/570
```

This memory usage summary displays allocations for each of the four major data structures used by **radiusd**, in the format x/y/z:

- **x** is the number of data structures allocated but not yet freed.
- **y** is the high-water mark (the most structures ever allocated but not freed at one time).
- **z** is the total number of structures allocated.

Bugs Fixed

- A misconfigured user entry that has a check item of Auth-Type = Local with no Password check item now rejects the user with the debug message **"Warning: entry for user <name> is missing its Password check item"**.
- An unknown Auth-Type check item now generates an error message and rejects the user.
- A memory leak that resulted from the use of multiple DEFAULT user entries is fixed.
- The password decryption code no longer calculates the next RSA Data Security, Inc. MD5 Message-Digest Algorithm (MD5) digest when it does not need to.
- The **radiusd** daemon is now strictly compliant with RFC 2139 and discards accounting-request packets with invalid request authenticators. As a result, you must run ComOS 3.3.1 or later to use RADIUS accounting with RADIUS server 2.1. The **-o** flag is provided for backwards compatibility with noncompliant RADIUS clients.
- Assorted memory leaks and pointer problems have been corrected.

RADIUS test program radtest

RADIUS 2.1 includes a simple example client program called **radtest**, that sends a RADIUS packet to a server running on the same host as **radtest**, and prints out the attributes returned. It doesn't support accounting packets yet. It always fills in the NAS-IP-Address as 127.0.0.1 and the NAS-Port as 1. Passwords longer than 16 characters are not supported. It looks for its dictionary in the same directory it is run from.

radtest -v prints the version.
radtest -h prints help:

```
./radtest -d called_id -f -g calling_id -h -i id -p port -s secret -t type -
v -x -u username password
```

The other flags work as follows:

-d Called Station Id

-f Send framed dialin hint

-g Calling Station Id

-i Use id as the packet identifier

-p Use port as the port (defaults to definition in **/etc/services** or 1645)

-s to specify shared secret (defaults to **localkey**)

-t send type as service type (overrides **-f**)

-u Specifies username and password (notice that this takes two arguments)

-x debug option (doesn't do anything currently)

How Proxy RADIUS Works and How to Configure It

Proxy RADIUS is a feature in which one RADIUS server can forward an authentication request to a remote RADIUS server, and return its reply to the NAS. A common use for proxy RADIUS is "roaming." Roaming permits two or more ISPs to allow each other's users to dial in to either ISP's network for service.

The network access server (NAS) (PortMaster) sends its RADIUS access-request to the forwarding server, which forwards it to the remote server. The remote server sends an access-accept, access-reject, or access-challenge response back to the forwarding server, which sends it back to the NAS. The choice of which server to forward the request to is based on the authentication realm.

Realms

A realm can be either of the following:

- The part following the "at" sign (@) in a username (a "named realm")
- A Called-Station-Id (a "numbered realm")

The forwarding server checks for a numbered realm before checking for a named realm. Frequently, a domain name is used as the named realm to provide uniqueness.

The RADIUS server 2.1 **radiusd** daemon also supports the realm/user style of username, but Lucent Remote Access recommends that you avoid this username style. Support is provided for older RADIUS servers that require it. However, because the "at" sign (@) always takes precedence over the slash (/), **radiusd** interprets the username "a/b@c" as user "a/b" in the named realm "c", for example. Such mixtures are strongly discouraged and might not be supported in future releases.

Accounting Information

RADIUS accounting-request packets are logged by both the forwarding server and remote server, but are acknowledged to the NAS only when the remote server sends an accounting-response back to the forwarding server.

Note – The remote server places the accounting information in a directory under **/usr/adm/radacct** named after the forwarding server, **not** the NAS.

Required Versions of RADIUS

Both the forwarding server and remote server must be running this release of the Lucent Remote Access RADIUS server, or a current version of Lucent PortAuthority™.

Any other vendor's conforming RADIUS proxy server is likely to work as either the forwarding server or remote server if that vendor has implemented proxy correctly. Lucent Remote Access RADIUS server versions 2.0.1 and earlier do not support proxy RADIUS, but can still be used as a remote server through the use of the **old** flag in the proxy file on the forwarding server.

Port Numbers Used

This RADIUS server listens on UDP port 1645 for access-requests and on UDP port 1646 for accounting-requests. It sends proxy requests from UDP ports 1650 and 1651 and listens for proxy responses on those ports. If the listening port is set with the -p flag to radiusd, then radiusd does the following:

- Listens on the specified UDP port for access-requests
- Listens on the port numbered 1 higher for accounting-requests
- Uses ports numbered 5 higher and 6 higher to send proxy requests

For example, if you run the **radiusd -p 1812** command, then **radiusd** uses UDP ports 1812, 1813, 1817 and 1818.

Versatility

The forwarding and remote RADIUS servers can run on different operating systems. A RADIUS server can function as both a forwarding server and a remote server, serving as a forwarding server for some realms and a remote server for other realms. Use care to avoid forwarding loops—a packet passed back and forth between two misconfigured forwarding servers. One forwarding server can forward to any number of remote servers (one per realm). A remote server can have any number of servers forwarding to it and can provide authentication for any number of realms.

Proxy Scenario

The following scenario illustrates the communication between a PortMaster and the forwarding and remote RADIUS servers:

1. A PortMaster sends its access-request to the forwarding server.
2. The forwarding server forwards the access-request to the remote server.
3. The remote server sends an access-accept, access-reject, or access-challenge back to the forwarding server. For this example, an access-accept is sent.
4. The forwarding server sends the access-accept to the PortMaster.
5. The PortMaster sends an accounting-request to the forwarding server.
6. The forwarding server logs the accounting-request and forwards it to the remote server.
7. The remote server logs the accounting-request and sends an accounting-response to the forwarding server.
8. The forwarding server sends the accounting-response to the PortMaster.

To set up proxy, create a proxy file in the **/etc/raddb** directory on the forwarding server. If named realms are used, a proxy file must also exist on the remote server. If only numbered realms are used, the remote server does not need a proxy file.

To use proxy, you set up RADIUS as you do normally. In addition, to form the communication between the forwarding and remote servers, you must define the following information in the clients and proxy files in **/etc/raddb**.

On the forwarding server:

- The clients file must have an entry for the PortMaster hostname or IP address and its shared secret.
- The proxy file must have an entry for the remote RADIUS server's hostname or IP address, its shared secret, and its realm. The shared secret in the forwarding server's proxy file must match the shared secret in the remote server's clients file.

On the remote server:

- The clients file must contain the forwarding server's hostname or IP address and its shared secret. The shared secret must match the shared secret in the forwarding server's proxy file.
- If any named realms are used, the proxy file must contain the hostname or IP address of the remote server itself, an unused shared secret, and the realm this remote server is authoritative for. If only numbered realms are used, then no proxy file needs to be defined on the remote server.

Proxy File Example

The **/etc/raddb/proxy** file contains proxy server hostnames (or IP addresses), shared secrets, and realms, all separated by spaces or tabs. Each line describes one realm. Here is a proxy file example:

```
radius.edu.net          secretupto16char      edu.net
s134.net.com            someothersec2ret      5551134
net54.edu.net           bettersecretthan      5555454
rad.edu.com             chsebetterth          edu.com 1645
rad7.com.net            lx4zDFapa3ep          com.net 1645 1646 old
radius.edu.net          eajsdfljasep          5551234 1812 1813 secure
eg.edu.net              e997asepdflj          edu.net old secure
```

- The first field is a valid hostname or IP address.
- The second field (separated by blanks or tabs) is the shared secret.
- The third field is the named or numeric authentication realm.
- The remaining fields can be empty, or can contain the RADIUS port number of the remote server, the RADIUS accounting port number of the remote server, and any of the following optional keywords:

  old Strips the realm from the username and does not attach Proxy-State when forwarding. This keyword is useful for forwarding requests to older RADIUS servers.

  secure Allows the remote server to authorize administrative logins for your client. If this keyword is not present, access-accepts from the forwarding server that grant Administrative or NAS-Prompt access are treated as access-rejects instead. If you use this keyword, you are allowing the remote server to let someone log in to your NAS as an administrator, so use it with caution!

  ipass Uses the iPass protocol (instead of RADIUS) to communicate with the remote server. See **http:/www.ipass.com/** for more information.

The optional fields can be specified in any order, separated by blanks or tabs, after the first three mandatory fields. If you specify a single UDP port, it is used for the RADIUS port. If you specify two ports, they are used as the RADIUS port and RADIUS accounting port in that order. If you specify no ports, they default to the same ports used by the RADIUS server itself.

If **secure** is not specified for a remote server and it replies with Service-Type = Administrative-User or NAS-Prompt-User, the forwarding server treats it as an access-reject and logs the following message to syslog:

```
Jul 10 21:10:00 ra radius[14870]: remote server 192.168.96.6/1645.4 returned
insecure service for client 172.16.3.24/1039.17, sending reject instead
```

If the hostname (or IP address) listed in the proxy file is the same as the primary hostname or IP address of the host running the RADIUS server, and the UDP port in the entry matches the UDP port the message was received on, **radiusd** determines that the user is local, strips off the "@domain" portion, and processes the request locally.

Special Named Realms

The special named realm DEFAULT (all uppercase) matches any named realm not found in the proxy file. If more than one DEFAULT entry exists in the proxy file, only the last one is used. For example:

```
center.com.net e199aespfdx4     DEFAULT
```

The special named realm NOREALM (all uppercase) matches any user that has no realm. If more than one NOREALM entry exists in the proxy file, only the last one is used. For example:

```
others.com.net e19aepsfd9x4     NOREALM
```

Example Configuration for Proxy RADIUS

The following example illustrates a typical proxy RADIUS topology and the sample configuration of proxy and clients files.

Equipment:

- PortMaster named **pmtest** with IP address 192.168.10.1
- Forwarding server named **forward** with an IP address of 192.168.10.2
- Remote server named **remote** with an IP address of 172.16.25.5

In a real configuration, you must use IP addresses or fully qualified domain names as hostnames.

1. Configure the contents of clients and proxy files of the server called “forward” as follows:

```
/etc/raddb/clients
------------------
pmtest            sharedsecret
/etc/raddb/proxy
----------------
remote            testsecret       com.net
```

2. Configure the contents of clients and proxy files of the server called **remote** as follows:

```
/etc/raddb/clients
------------------
forward           testsecret
/etc/raddb/proxy
----------------
remote            doesntmatter     com.net
```

3. On the PortMaster **pmtest**, enter the following commands to set the authentication and accounting servers:

```
set authentic 192.168.10.2
set secret sharedsecret
set accounting 192.168.10.2
save all
```

4. For a user to be authenticated via the remote server, define a profile for this user in the users file of the remote server. A user profile is defined in the following format. Note that the remote server strips the named realm from the username before looking it up in the users file.

```
test      Password = "testing"
          Service-Type = Framed-User,
          Framed-Protocol = PPP,
          Framed-IP-Address = 255.255.255.254,
          Framed-Routing = None
```

   Alternatively, if the test user's password is stored in the **/etc/passwd** file, the example user profile is the following:

```
test        Auth-Type = System
            Service-Type = Framed-User,
            Framed-Protocol = PPP,
            Framed-IP-Address = 255.255.255.254,
            Framed-Routing = None
```

5. Run the **radiusd** daemon on both **forward** and **remote** servers. The RADIUS accounting records are logged in the detail file of each server.

   A user dialing in to the PortMaster must enter **test@com.net** at the login prompt and a password at the password prompt.

Limitation of Proxy

For the RADIUS server to handle numbered realms for points of presence (POPs) from multiple area codes, the RADIUS server must be configured with the area code of each PortMaster if that information is not included in the Called-Station-Id. The ability to determine the area code is not included in RADIUS server 2.1.

This limitation is a problem only if your situation includes **all** of the following:

- You use the same seven-digit telephone number in multiple area codes to belong to different realms.
- You use the same RADIUS forwarding server for all area codes and/or all realms.
- Your telephone company does not include the area code in the Calling-Station-Id.

Upgrading

RADIUS server 2.1 is available in source form at **ftp://ftp.livingston.com/pub/le/radius/radius21.tar.Z** and in binary form for the following platforms at **ftp://ftp.livingston.com/pub/le/software/:**

- IBM RS6000 AIX 4.2aix/radius21.tar.Z
- Alpha Digital UNIX 4.0alpha/radius21.tar.Z
- BSD/OS 3.0bsdi/radius21.tar.Z

- HP/UX 10.20hp/radius21.tar.Z
- Slackware Linux 2.0linux/radslack21.tar.Z
- Redhat Linux 5.2linux/radhat21.tar.Z
- SGI IRIX 6.3sgi/radius21.tar.Z
- SunOS 4.1.4sun4/radsun21.tar.Z
- Solaris 2.5.1sun4/radsol21.tar.Z
- Solaris x86 2.5.1sun86/radius21.tar.Z

For other flavors of UNIX, including Linux, FreeBSD, NetBSD, and BSD/OS 4.0, get the source and compile from that. RADIUS 2.1 is not available for Windows NT.

To upgrade, do the following:

1. Save a copy of your old dictionary file and **radiusd** daemon.
2. Copy the new dictionary file to **/etc/raddb** or whatever directory you use.
3. If you are using proxy, create a **/etc/raddb/**proxy file.
   - Kill your existing radiusd, and run the new **radiusd**.
     If you are using SecurID or ActivCard for authentication, run **sradiusd** instead of **radiusd**.
   - If you are using iPass, run **iradiusd** instead of **radiusd**.
   - If you are running iPass AND SecurID or ActivCard, modify the **Make** file to link all the appropriate libraries. Contact **support@livingston.com** if you need assistance in doing so.

Copyright and Trademarks

PortMaster, ComOS, and ChoiceNet are registered trademarks of Lucent Technologies Inc. PMVision, IRX, and PortAuthority are trademarks of Lucent Technologies Inc. PolicyFlow is a service mark of Lucent Technologies Inc. All other marks are the property of their respective owners.

Notices

Lucent Technologies Inc. makes no representations or warranties with respect to the contents or use of this publication, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Lucent Technologies, Inc. reserves the right to revise this publication and to make changes to its content, any time, without obligation to notify any person or entity of such revisions or changes.

Contacting Lucent Remote Access Technical Support

Lucent Technologies Remote Access Business Unit (previously Livingston Enterprises) provides technical support via voice, electronic mail, or through the World Wide Web at **http://www.livingston.com/**.

Please include the output of radiusd **-v** and uname **-a** when reporting problems with this release.

Internet service providers (ISPs) and other end users in Europe, the Middle East, Africa, India, and Pakistan should contact their authorized Lucent Remote Access sales channel partner for technical support; see **http://www.livingston.com/International/EMEA/distributors.html**.

For North and South America and Asia Pacific customers, technical support is available Monday through Friday from 7 a.m. to 5 p.m. U.S. Pacific Time (GMT -8). Dial 1-800-458-9966 within the United States (including Alaska and Hawaii), Canada, and the Caribbean and Latin America (CALA), or 1-925-737-2100 from elsewhere, for voice support. For email support send to support@livingston.com (**asia-support@livingston.com** for Asia Pacific customers).