Client Upgrade Procedures Guide for HP OpenView Using Radia

for UNIX operating systems

Software Version: 4.1

Client Upgrade Procedures Guide

June 2005

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

Trademark Notices

Linux is a registered trademark of Linus Torvalds.

Microsoft®, Windows®, and Windows® XP are U.S. registered trademarks of Microsoft Corporation.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Acknowledgements

PREBOOT EXECUTION ENVIRONMENT (PXE) SERVER
Copyright © 1996-1999 Intel Corporation.

Support

Please visit the HP OpenView web site at:

http://www.managementsoftware.hp.com/

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

You can also go directly to the support web site at:

http://support.openview.hp.com/

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

http://support.openview.hp.com/access_level.jsp

To register for an HP Passport ID, go to:

https://passport2.hp.com/hpp/newuser.do

Contents

1 Upgrading Your Radia Clients.. 7

About the Radia Client Upgrade 7

Upgrading the Radia Client for Unix as a Radia Service 7

Importing the Export Decks 8
The UPGRADE Class 14
ZSTOP Expressions 14
Always Connection 14
Create Method (ZCREATE) 15
Verify Method (ZVERIFY) 15
Update Method (ZUPDATE)..15
Media Directory (MEDIADIR)..16
Retry Attributes (ATTEMPTS and INTERVAL)..16
Command Line (CMDLINE)..16
Configuring the Radia Client Upgrade..18
To configure the Radia Client Upgrade 19
Customizing The Radia Client Upgrade Services..21
Adding the UPGRADE Class to the Master Connection Rules 22
Mechanics of the Radia Client Upgrade for Unix when applied as a Radia Service...28
Troubleshooting the Radia Client Upgrade for Unix..29
Upgrading the Radia Client to Version 3.x – from CDROM 29

1 Upgrading Your Radia Clients

Use the information in this chapter to upgrade your AIX, RedHat™ Linux, HP-UX, and Sun Solaris (Sparc architecture) Radia Clients to version 4.1.

Please see the Radia Application Manager Guide and Radia Software Manager Guide for additional details.

About the Radia Client Upgrade

Novadigm offers several ways to upgrade your Radia clients:

- Using the Radia Client Upgrade service (*recommended*)
  Use the Radia Client Upgrade service to distribute the client upgrade as a Radia Service.
- Using the standard installation media
  Use the Client Upgrade installation media if you want to upgrade a machine in your lab environment. The Radia 4.x Client upgrade contains the standard installation files, as well as an upgrade file. When you run the upgrade, a standard installation is performed; or, if it detects the existence of a previous client version, it launches the upgrade. Existing client object attributes are maintained and the client version information is updated to reflect version 4.x.
- For information about the standard installation of Radia clients, see the Unix version of the *Radia Application Manager Guide*, *Radia Inventory Manager Guide* or *Radia Software Manager Guide*.

Upgrading the Radia Client for Unix as a Radia Service

This section describes how to perform a Radia client upgrade for Unix using the following export decks.

- UPGRADE.XPC
  This is a class export of a new class needed for the Unix Radia client

upgrade. It contains the definition of the PRIMARY.SOFTWARE.UPGRADE class

- Client Operating System specific Instance and Resource client upgrade materials

  For IBM/AIX

  — AIX_41_RADIA_CLIENT_UPGRADE.XPI

  — AIX_41_RADIA_CLIENT_UPGRADE.XPR

  For RedHat™ Linux:

  — LINUX_41_RADIA_CLIENT_UPGRADE.XPI

  — LINUX_41_RADIA_CLIENT_UPGRADE.XPR

  For HP-UX:

  — HP_41_RADIA_CLIENT_UPGRADE.XPI

  — HP_41_RADIA_CLIENT_UPGRADE.XPR

  For Solaris:

  — SOL_41_RADIA_CLIENT_UPGRADE.XPI

  — SOL_41_RADIA_CLIENT_UPGRADE.XPR

  For Solaris x86:

  — SOLX86_41_RADIA_CLIENT_UPGRADE.XPI

  — SOLX86_41_RADIA_CLIENT_UPGRADE.XPR

Importing the Export Decks

In order to distribute the Radia client upgrade as a Radia service, you must import the Radia Client **UPGRADE** class definition as well as upgrade service data into your Radia Database.

To import the export decks

The examples below use the HP-UX specific export deck filenames after the ZEDMAMS command line argument `FILE=` in steps 7 and `10.` When performing the steps outlined below be sure to use the filenames (prefixed with the target OS platform) pertaining to the specific operating systems of interest to you in your enterprise. Specify the appropriate operating system export decks as defined above.

1 Stop the Radia Configuration Server.

2 Transfer the appropriate export decks (as listed above) into the Radia Configuration Server binary/executable directory.

SystemDrive:\Radia\ConfigurationServer\bin, if your Radia Configuration Server is running on a Microsoft Windows platform.

OR

…./Novadigm/ConfigurationServer/exe, if your Radia Configuration Server is running on a Unix platform

If you need to ftp the XPI and XPR files from one machine to another, XPR files are binary and should be transferred in binary mode, while XPI files are ASCII and should be transferred in ASCII mode.

3 Change your current working directory to the Radia Configuration Server binary/executable directory where you just copied the export decks and type:

```
ZEDMAMS VERB=IMPORT_CLASS,FILE=UPGRADE.XPC,PREVIEW=NO
```

4 Press ENTER.

5 Review the zedmams.log to ensure that there were no errors, and that your import completed with RC 0.

6 After the UPGRADE class has been imported, type:

```
ZEDMAMS
VERB=IMPORT_INSTANCE,FILE=HP_41_RADIA_CLIENT_UPGRADE.XPI,PREVIE
W=NO
```

7. Press ENTER.
8. Review the zedmams.log to ensure that there were no errors, and that your import completed with RC 0.
9. After the instances have been imported, type:

   ```
   ZEDMAMS
   VERB=IMPORT_RESOURCE,FILE=HP_41_RADIA_CLIENT_UPGRADE.XPR,PREVIE
   W=NO
   ```

10. Press ENTER.
11. Review the zedmams.log to ensure that there were no errors, and that your import completed with RC 0.

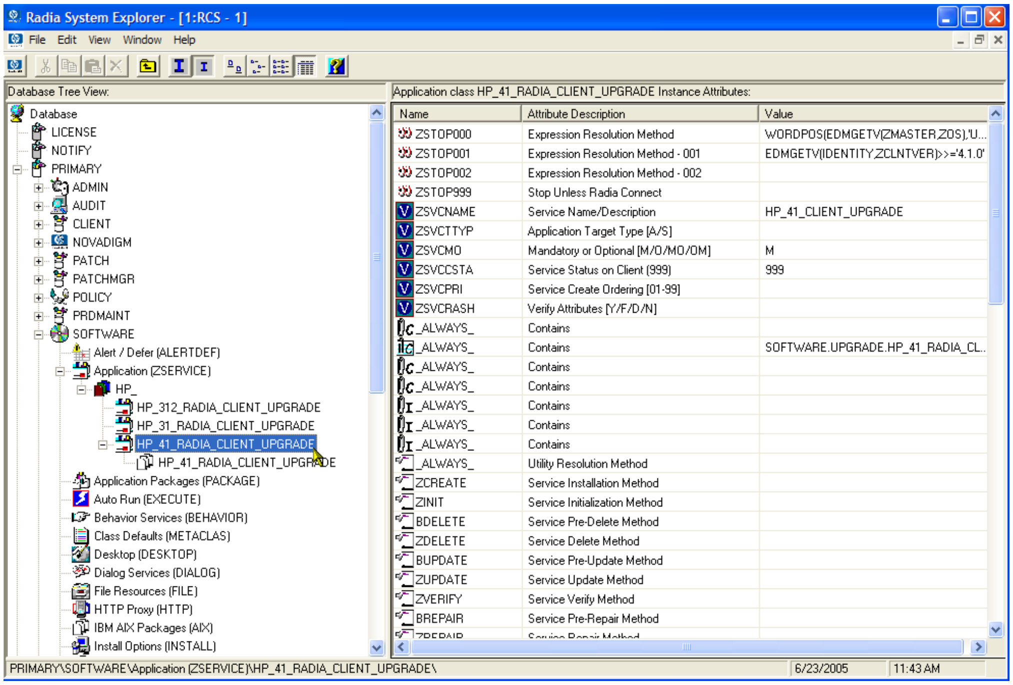
    Repeat the steps above until each of the Operating System specific XPI and XPR files relevant to your enterprise needs have been imported into your Radia Configuration Server database. Then, start the Radia Configuration Server and use the Radia System Explorer to view the new UPGRADE class and service specific data.

Upon the successful completion of the above steps, you will find:

- A new class called **UPGRADE** within the PRIMARY.SOFTWARE domain.
- Sample Unix specific Client upgrade Application Service(s) within your SOFTWARE.ZSERVICE class.
- Sample UPGRADE instance data within the PRIMARY.SOFTWARE.UPGRADE class.
- Sample PACKAGE(s) within to the PRIMARY.SOFTWARE domain.

These new Service(s) imported into the Radia Configuration Server database are Mandatory Services (to be distributed to end clients via the Radia Application Manager).

The following figure shows the Novadigm supplied Unix Client upgrade services as shown through a Radia System Explorer.



Database Tree View:

- Database
  - LICENSE
  - NOTIFY
  - PRIMARY
    - ADMIN
    - AUDIT
    - CLIENT
    - NOVADIGM
    - PATCH
    - PATCHMGR
    - POLICY
    - PRDMAINT
    - SOFTWARE
      - Alert / Defer (ALERTDEF)
      - Application (ZSERVICE)
        - HP_
          - HP_312_RADIA_CLIENT_UPGRADE
          - HP_31_RADIA_CLIENT_UPGRADE
          - HP_41_RADIA_CLIENT_UPGRADE
            - HP_41_RADIA_CLIENT_UPGRADE
      - Application Packages (PACKAGE)
      - Auto Run (EXECUTE)
      - Behavior Services (BEHAVIOR)
      - Class Defaults (METACLAS)
      - Desktop (DESKTOP)
      - Dialog Services (DIALOG)
      - File Resources (FILE)
      - HTTP Proxy (HTTP)
      - IBM AIX Packages (AIX)
      - Install Options (INSTALL)

Application class HP_41_RADIA_CLIENT_UPGRADE Instance Attributes:

Name	Attribute Description	Value
ZSTOP000	Expression Resolution Method	WORDPOS(EDMGETV(ZMASTER,ZOS),'U...
ZSTOP001	Expression Resolution Method - 001	EDMGETV(IDENTITY,ZCLNTVER)>>='4.1.0'
ZSTOP002	Expression Resolution Method - 002	
ZSTOP999	Stop Unless Radia Connect	
ZSVCNAME	Service Name/Description	HP_41_CLIENT_UPGRADE
ZSVCTTYP	Application Target Type [A/S]	
ZSVCMO	Mandatory or Optional [M/O/MO/OM]	M
ZSVCCSTA	Service Status on Client (999)	999
ZSVCPRI	Service Create Ordering [01-99]	
ZSVCRASH	Verify Attributes [Y/F/D/N]	
ALWAYS	Contains	
ALWAYS	Contains	SOFTWARE.UPGRADE.HP_41_RADIA_CL...
ALWAYS	Contains	
ALWAYS	Contains	
ALWAYS	Contains	
ALWAYS	Contains	
ALWAYS	Contains	
ALWAYS	Utility Resolution Method	
ZCREATE	Service Installation Method	
ZINIT	Service Initialization Method	
BDELETE	Service Pre-Delete Method	
ZDELETE	Service Delete Method	
BUPDATE	Service Pre-Update Method	
ZUPDATE	Service Update Method	
ZVERIFY	Service Verify Method	
BREPAIR	Service Pre-Repair Method	
ZREPAIR	Service Repair Method	

PRIMARY\SOFTWARE\Application (ZSERVICE)\HP_41_RADIA_CLIENT_UPGRADE\   6/23/2005   11:43 AM

Figure 1: Radia client for Unix Upgrade specific Services

The supplied Radia Client Upgrade Application Service(s) are modeled per the figure shown below:

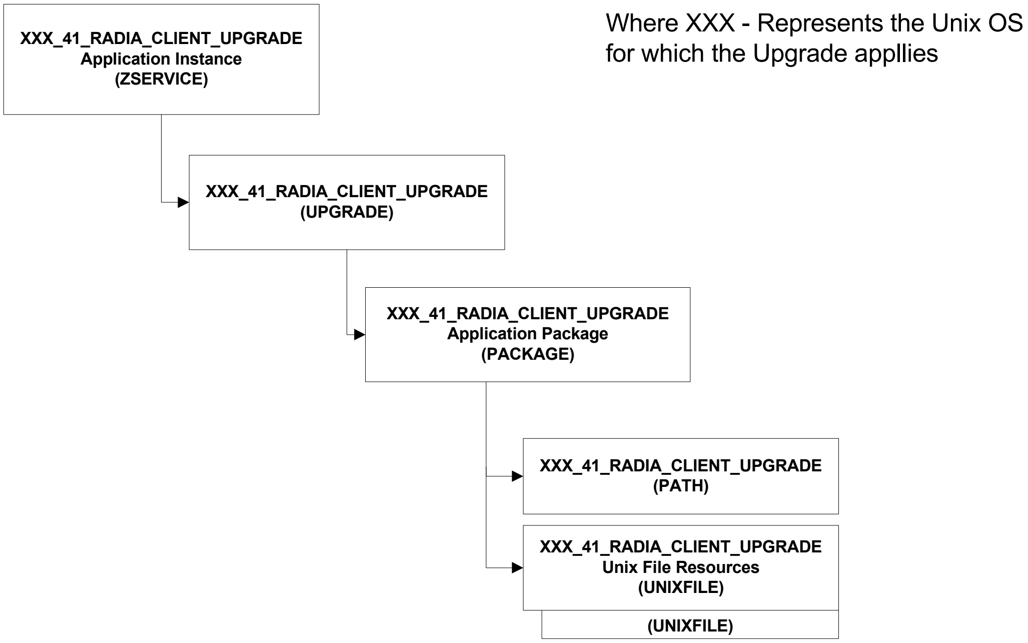

Where XXX - Represents the Unix OS for which the Upgrade appllies
XXX_41_RADIA_CLIENT_UPGRADE
Application Instance
(ZSERVICE)
XXX_41_RADIA_CLIENT_UPGRADE
(UPGRADE)
XXX_41_RADIA_CLIENT_UPGRADE
Application Package
(PACKAGE)
XXX_41_RADIA_CLIENT_UPGRADE
(PATH)
XXX_41_RADIA_CLIENT_UPGRADE
Unix File Resources
(UNIXFILE)
(UNIXFILE)

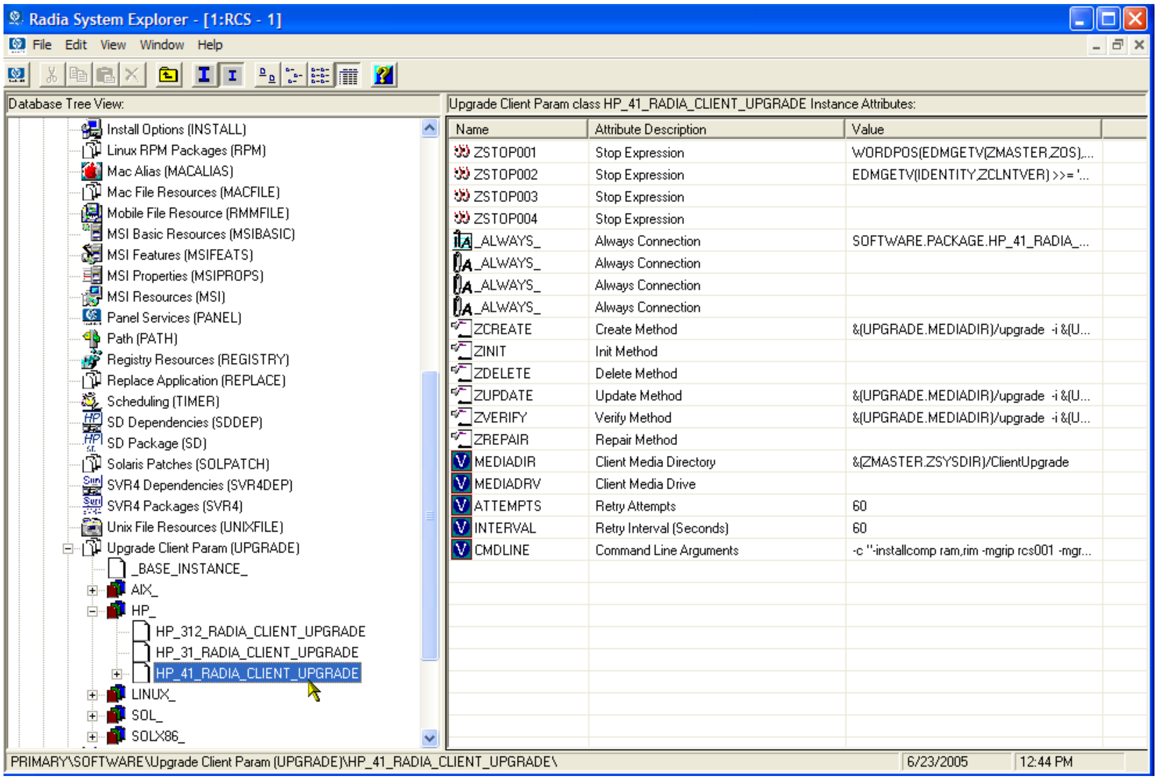
Figure 2~ Model for the Unix Client Upgrade Service

Novadigm provides the following Sample Services which can be used to perform an upgrade of Radia Version 2.X or Version 3.0 Clients to the current Radia Version 3.1. The sample Service names provided are called:

- AIX_41_RADIA_CLIENT_UPGRADE
  Installation of this Service would upgrade a pre-existing IBM AIX Radia client to the current Version 4.1 level.
- HP_41_RADIA_CLIENT_UPGRADE
  Installation of this Service would upgrade a pre-existing HP-UX (hprisc architecture) Radia client to the current Version 4.1 level.
- LINUX_41_RADIA_CLIENT_UPGRADE
  Installation of this Service would upgrade a pre-existing Red Hat Linux (Intel architecture) Radia client to the current Version 4.1 level.
- SOL_41_RADIA_CLIENT_UPGRADE
  Installation of this Service would upgrade a pre-existing Sun Solaris (sparc architecture) Radia client to the current Version 4.1 level.

- SOLX86_41_RADIA_CLIENT_UPGRADE
  Installation of this Service would upgrade a pre-existing Sun Solaris (X86 architecture) Radia client to the current Version 4.1 level.

All of the configurable parameters pertaining to the Radia Client Upgrade are specified in the UPGRADE instance in the Radia Configuration Server entitlement model. The figure shown below shows a sample Unix Client UPGRADE instance.


Radia System Explorer - [1:RCS - 1]
File
Edit
View
Window
Help
Database Tree View:
Install Options (INSTALL)
Linux RPM Packages (RPM)
Mac Alias (MACALIAS)
Mac File Resources (MACFILE)
Mobile File Resource (RMMFILE)
MSI Basic Resources (MSIBASIC)
MSI Features (MSIFEATS)
MSI Properties (MSIPROPS)
MSI Resources (MSI)
Panel Services (PANEL)
Path (PATH)
Registry Resources (REGISTRY)
Replace Application (REPLACE)
Scheduling (TIMER)
SD Dependencies (SDDEP)
SD Package (SD)
Solaris Patches (SOLPATCH)
SVR4 Dependencies (SVR4DEP)
SVR4 Packages (SVR4)
Unix File Resources (UNIXFILE)
Upgrade Client Param (UPGRADE)
_BASE_INSTANCE_
AIX_
HP_
HP_312_RADIA_CLIENT_UPGRADE
HP_31_RADIA_CLIENT_UPGRADE
HP_41_RADIA_CLIENT_UPGRADE
LINUX_
SOL_
SOLX86_

Upgrade Client Param class HP_41_RADIA_CLIENT_UPGRADE Instance Attributes:

Name	Attribute Description	Value
ZSTOP001	Stop Expression	WORDPOS(EDMGETV(ZMASTER,ZOS),...
ZSTOP002	Stop Expression	EDMGETV(IDENTITY,ZCLNTVER) >>= '...
ZSTOP003	Stop Expression	
ZSTOP004	Stop Expression	
ALWAYS	Always Connection	SOFTWARE.PACKAGE.HP_41_RADIA_...
ALWAYS	Always Connection	
ALWAYS	Always Connection	
ALWAYS	Always Connection	
ZCREATE	Create Method	&(UPGRADE.MEDIADIR)/upgrade -i &(U...
ZINIT	Init Method	
ZDELETE	Delete Method	
ZUPDATE	Update Method	&(UPGRADE.MEDIADIR)/upgrade -i &(U...
ZVERIFY	Verify Method	&(UPGRADE.MEDIADIR)/upgrade -i &(U...
ZREPAIR	Repair Method	
MEDIADIR	Client Media Directory	&(ZMASTER.ZSYSDIR)/ClientUpgrade
MEDIADRV	Client Media Drive	
ATTEMPTS	Retry Attempts	60
INTERVAL	Retry Interval (Seconds)	60
CMDLINE	Command Line Arguments	-c "-installcomp ram,rim -mgrip rcs001 -mgr...

PRIMARY\SOFTWARE\Upgrade Client Param (UPGRADE)\HP_41_RADIA_CLIENT_UPGRADE\ 6/23/2005 12:44 PM

Figure 3: Radia client for Unix upgrade Instance

The UPGRADE Class

The UPGRADE class instance is connected between the upgrade SERVICE and PACKAGE instances. It is responsible for the resolution of the client upgrade materials as well as for launching the client upgrade process. All upgrade configuration options are defined within the UPGRADE class instance.

The instance names and instance connections named in this section of the document uses the Solaris Upgrade Service and Upgrade Instance data for the purpose of providing sample attribute data.

ZSTOP Expressions

The sample UPGRADE instance(s) contain two ZSTOP expressions, one for the specific operating system for which the UPGRADE applies and one for the Radia Application Manager client version. The purpose of the expressions is to gate the resolution for a specific operating system and to prevent the resolution for an already upgraded client.

For example, in the SOFTWARE.UPGRADE.HP_41_RADIA_CLIENT_UPGRADE instance, the first ZSTOP expression “WORDPOS(EDMGET(ZMASTER.ZOS),’UNIXHPUX’)=0” causes the resolution to continue only if the client operating system is HP-UX.

The second ZSTOP expression “EDMGETV(IDENTITY.ZCLNTVER)>>=’4.1.0’ “ causes the resolution to continue only if the ZCLNTVER variable in the current Radia Application Managers client IDENTITY object contains a value less than 4.1.0. This expression would stop the deployment of the Radia client upgrade to Radia clients which are already running Radia Version 4.1.

These same expressions can be found in the sample SERVICE instances provided.

Always Connection

The always connection in the upgrade instance creates an association with the operating system specific package which contains the client upgrade materials. (for example, the SOFTWARE.UPGRADE.HP_41_RADIA_CLIENT_UPGRADE instance has an ALWAYS connection of:

SOFTWARE.PACKAGE.HP_41_RADIA_CLIENT_UPGRADE, the package which contains the HP-UX Client Upgrade materials). The value of this attribute in any UPGRADE instance should contain the name of the PACKAGE instance that is prefixed with the Operating System specific designator that matches the intended target operating system.

Create Method (ZCREATE)

The sample UPGRADE instance contains the create method value of "&(UPGRADE.MEDIADIR)/upgrade –I &(UPGRADE.INTERVAL) –a &(UPGRADE.ATTEMPTS) &(UPGRADE.CMDLINE) &". The value, specified in the create method attribute, causes the upgrade script to be launched with the values specified in the UPGRADE class instance attributes. This method is called only when the client upgrade materials are initially installed on disk.

Verify Method (ZVERIFY)

The sample UPGRADE instance contains the verify method value of "&(UPGRADE.MEDIADIR)/upgrade –I &(UPGRADE.INTERVAL) –a &(UPGRADE.ATTEMPTS) &(UPGRADE.CMDLINE) &". The value, specified in the verify method attribute, causes the upgrade script to be launched with the values specified in the UPGRADE class instance attributes. This method is only called after a successful verify. If the client upgrade was not able to complete successfully when the upgrade materials were installed due to disk space, client in use, or for any reason, this would initiate the upgrade process again.

Update Method (ZUPDATE)

The sample UPGRADE instance contains the update method value of "&(UPGRADE.MEDIADIR)/upgrade –I &(UPGRADE.INTERVAL) –a &(UPGRADE.ATTEMPTS) &(UPGRADE.CMDLINE) &". The value, specified in the update method attribute, causes the upgrade script to be launched with the values specified in the UPGRADE class instance attributes. This method is called after a successful update. If the client upgrade was not able to complete successfully when the upgrade materials were first installed and updated client upgrade materials are available in the connecting Radia Configuration Server, the update method would be launched after the successful installation of those updated materials, initiating the upgrade process again.

Media Directory (MEDIADIR)

The media directory attribute “MEDIADIR” contains the value of “&(ZMASTER.ZSYSDIR)/ClientUpgrade”. This is the default location where the upgrade process expects to find the Client upgrade materials, when they are downloaded to the client workstation.

Retry Attributes (ATTEMPTS and INTERVAL)

When the Radia client upgrade service finds the Radia client is in use, it will retry the upgrade a number of times at a specified interval. The "ATTEMPTS" attribute sets the number of times to retry the upgrade, while the "INTERVAL" attribute specifies the number of seconds the upgrade waits between retry attempts. The default values for both the “ATTEMPTS” and “INTERVAL” attributes is 60.

A value of zero in the "ATTEMPTS" attribute will cause the upgrade process to retry indefinitely.

Command Line (CMDLINE)

The command line attribute, “CMDLINE”, is used to specify the command line arguments to be passed to the client upgrade process. Because all arguments specified after the –c option are passed directly to the client installer, any arguments that can be specified on the client install command line can be specified here except for the “**–mode**” argument which is automatically set by the upgrade script to perform a silent install/upgrade.

The leading text “–c “ in the command line attribute called “CMDLINE” is required.

The command line arguments following the –c option must be placed inside double quotes.

The **–installcomp** command line option is used to specify the specific products to be upgraded/installed on the client workstation.

Valid client components include:

ram	- specifies Radia Application Manager
rim	- specifies Radia Inventory Manager

rsm - specifies Radia Software Manager

For example, to upgrade your Radia Application Manager and also install the Radia Inventory Manager and Radia Software Manager, you would specify “-installcomp ram,rim,rsm”.

When multiple products are specified the Radia product acronyms must be comma separated.

The **–mgrip** and –**mgrport** command line options specify the Radia Configuration Server ip address and port that the upgraded client will use for it’s ZMASTER object and sample shell scripts. Should you upgrade or install the Radia Software Manager the Software Manager’s configuration file called args.xml would be populated with Radia Configuration Server IP address and port number specified with the **–mgrip** and **–mgrport** command line options. These parameter values do not have to be the same value as the Radia Configuration Server IP and port, from which the client is receiving the upgrade service. The default value for –mgrport is 3464.

Sample UPGRADE instances specify a CMDLINE attribute value of: -c “-instcomp ram,rim –mgrip rcs001–mgrport 3464” which would upgrade Radia Application Manager and Radia Inventory Manager client components and pre-configure the upgraded client to receive new applications from the Radia configuration server called rcs001 running on port 3464.

Therefore please be sure to update the value of the UPGRADE instance “CMDLINE” attribute to reflect your preferred Radia Configuration Server ip and port.

You may use symbolic substitution for the values passed via the **–mgrip** and **–mgrport** command line arguments, providing the Radia Configuration Server from which the client receives the Client Upgrade Service is the same as Radia Configuration Server from which you would like the upgraded client receive future Application Services. For example the UPGRADE.CMDLINE attribute may be configured to read:

-c “-instcomp ram,rim,rsm –mgrip &(ZMASTER.ZIPADDR) –mgrport &(ZMASTER.ZDSTSOCK)

This would upgrade Radia Application Manager, Radia Inventory Manager and Radia Software Manager on the client receiving the service and pre-configure the upgraded client to receive new applications from the Radia configuration server from which it installed the Radia Client Upgrade Service.

Minimally, we recommend the use of the following upgrade command line options **–instcomp, –mgrip and –mgrport.**

You can append the text **–loglevel 9** in the command line to set maximum logging for the client upgrade/install. This command line option is typically used for diagnostic purposes.

Alternatively, you may specify a Command Line to use a Client Install configuration file.

For information about installer configuration files (created by the Remote Installation Setup process of the Radia client installer), please consult the Client installation chapter of a Radia client guide for Unix.

Should you have an installation configuration file already on the Client workstations, (created by the Remote Installation Setup process of the Radia client installer), you can have the client upgrade use this configuration file as a basis to perform a client upgrade by specifying **–cfg configfilename.cfg** in the CMDLINE attribute, where configfilename.cfg, specifies a fully qualified path name to a configuration file.

It is not recommended to specify both a configuration file and command line options **–installcomp**, **–mgrport** or **–mgrip** because the command line options will override the values specified in the configuration file.

For example, the command line specification of –c " –loglevel 9 –cfg /tmp/myown.cfg" would rely exclusively on the variables set in the /tmp/myown.cfg configuration file to determine what Radia components to upgrade/install and what Radia Configuration Server IP and port to use as well as other upgrade/install options.

Configuring the Radia Client Upgrade

The following example uses the HP-UX operating system files. Make sure to substitute your appropriate operating system designator when necessary.

To configure the Radia Client Upgrade

1 In the Radia System Explorer, navigate to the Novadigm supplied Application Service pertinent to the target Operating System for which you intend to apply the client upgrade, for example, **SOFTWARE.ZSERVICE.HP_41_RADIA_CLIENT_UPGRADE**. The Service instance is connected to an UPGRADE class instance which contains the configuration options relevant to the client upgrade.

2 Go to the UPGRADE instance (PRIMARY.SOFTWARE.UPGRADE) for example, **HP_41_RADIA_CLIENT_UPGRADE**.

3 In the right-hand pane double-click **CMDLINE**. This allows you to define specific Client Upgrade options and behaviors. This attribute is used to specify the Radia client components you wish to upgrade/install as well as your preferred Radia Configuration Server ip address and port number.

4 In the **Editing Instance** dialog box, edit the value to include all the client components you would like to upgrade as well as your preferred Radia Configuration Server ip address and port number.

Valid Client components include:

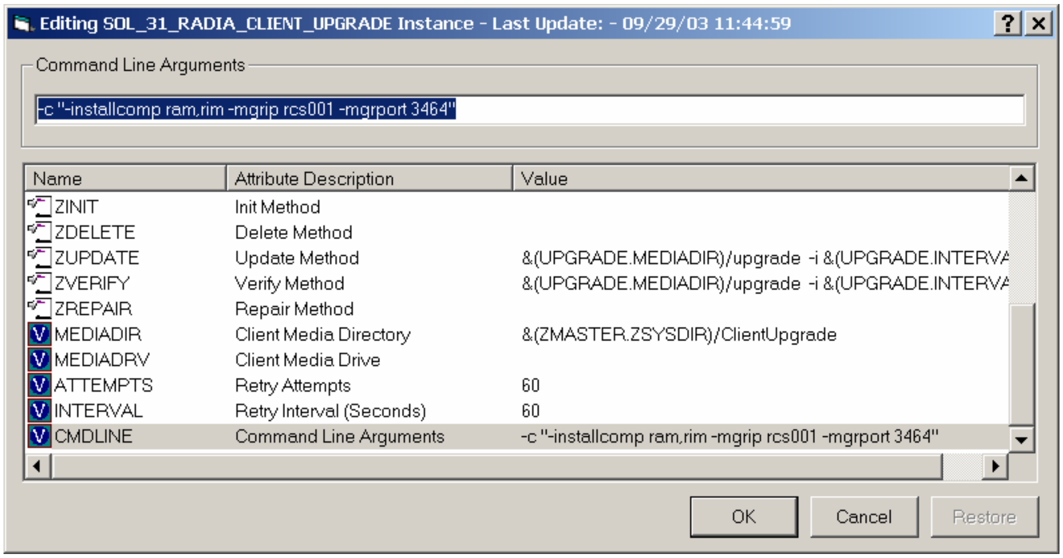
ram	- specifies Radia Application Manager
rim	- specifies Radia Inventory Manager
rsm	- specifies Radia Software Manager

The command line arguments following the –c option must be placed inside double quotes.

When multiple client components are specified the Radia product acronyms must be comma separated.

For example, to specify upgrades for the **Radia Application Manager, and Radia Inventory Manager** and have these upgraded client components configured to connect to a Radia Configuration Server called **rcs001** running on port **3464**, your CMDLINE (command line) attribute would look like the following (and shown in the Figure below):

```
-c “installcomp ram,rim –mgrip rcs001 –mgrport 3464”
```



Editing SOL_31_RADIA_CLIENT_UPGRADE Instance - Last Update: - 09/29/03 11:44:59

Command Line Arguments

-c "-installcomp ram,rim -mgrip rcs001 -mgrport 3464"

Name	Attribute Description	Value
ZINIT	Init Method	
ZDELETE	Delete Method	
ZUPDATE	Update Method	&(UPGRADE.MEDIADIR)/upgrade -i &(UPGRADE.INTERVA
ZVERIFY	Verify Method	&(UPGRADE.MEDIADIR)/upgrade -i &(UPGRADE.INTERVA
ZREPAIR	Repair Method	
V MEDIADIR	Client Media Directory	&(ZMASTER.ZSYSDIR)/ClientUpgrade
V MEDIADRV	Client Media Drive	
V ATTEMPTS	Retry Attempts	60
V INTERVAL	Retry Interval (Seconds)	60
V CMDLINE	Command Line Arguments	-c "-installcomp ram,rim -mgrip rcs001 -mgrport 3464"

5 Click **OK**, and then click **Yes** to confirm your changes.

6 If you would like to alter the values of the number of times the client will attempt to install the client upgrade as well as the time interval between attempts, change the values of the **ATTEMPTS** and **INTERVAL** attributes in the same manner. The remaining attributes should retain their default values for best performance.

7 Connect the OS specific sample service (e.g. HP_41_RADIA_CLIENT_UPGRADE) to the appropriate users/machines for distribution. See the *Radia Application Manager Guide for Unix* or the *Radia Software Manager Guide for Unix* for more information.

Before connecting an upgrade service to any client instances, make sure any users/machines that are configured to receive the Radia client upgrade Service are not connected to self-maintenance for pre Version 4.1 clients. If self-maintenance is not disconnected and the upgrade is installed and completed, you could run the risk of back-leveling the upgraded client by the application of client maintenance for earlier versions of the Radia clients on the client's next connect to the Radia Configuration Server.

After the upgrade service is distributed to a user and the version 3.x upgrade process is complete, when the Client connects to the Radia Configuration Server again, the upgrade service will no longer be available to the user/machine as it is automatically removed from the desktop in order to save disk space. This is a result of the ZSTOP expression within the upgrade instance evaluating to TRUE after the client is successfully upgraded to version 3.x. Once this expression evaluates to TRUE, the temporary service folder &(ZMASTER.ZSYSDIR)/ClientUpgrade and the client installation files are removed from the client desktop.

The ZSTOP expressions in the sample Upgrade Application Service (ZSERVICE) instances as well as the supplied UPGRADE instances are OS specific, these expressions are there to ensure the appropriate Client upgrade materials are distributed to clients that match the Operating System identifier contained in the expression.

Customizing The Radia Client Upgrade Services

Should you decide to create your own client upgrade Application Service (ZSERVICE instance) to meet your enterprise needs, Novadigm strongly recommends that you do so by copying one of the supplied sample instances to a new name, ensuring that the instance prefix (used to denote the target operating system) that is used as the source of the copy, matches your intended target operating system. This is to ensure that your new instance has the necessary ZSTOP expression pertinent to the Operating System to which the upgrade applies.

If you decide to customize UPGRADE instance data, to meet your enterprise needs, Novadigm strongly recommends that you do so by copying one of the supplied sample instances to a new name ensuring that the instance prefix (used to denote the target operating system) that is used as the source of the copy matches your intended target operating system. This is to ensure that your new instance has the necessary ZSTOP expression pertinent to the Operating System to which the upgrade applies.   By performing this copy instance function within the UPGRADE class your custom UPGRADE instance should already have a connection to the appropriate SOFTWARE.PACKAGE instance for your intended target operating system. You should then, connect your custom UPGRADE instance to your custom

ZSERVICE instance replacing the pre-existing UPGRADE instance connection in the Application Service instance.

Always be certain your Unix Client Upgrade Service has connections like that depicted in Figure 2 of this document.

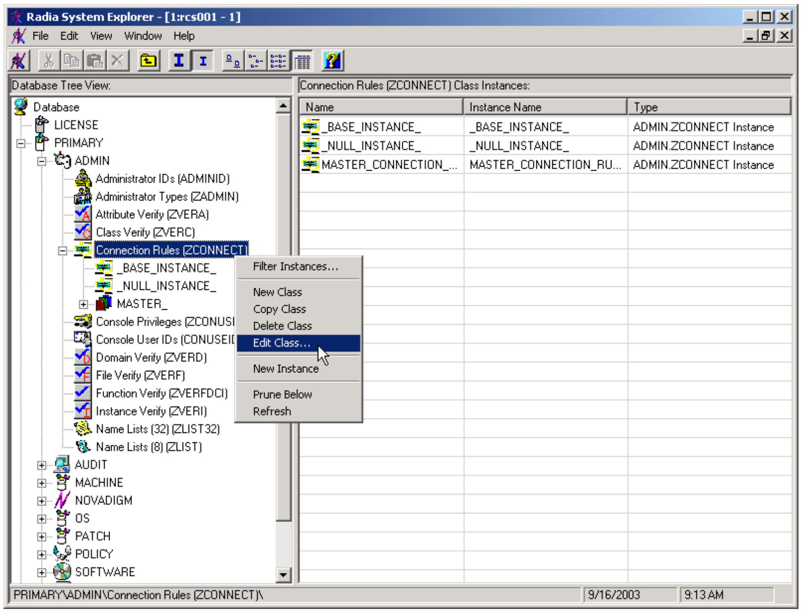
Adding the UPGRADE Class to the Master Connection Rules

Should it be necessary to add new ZSERVICE/UPGRADE instances for the client upgrade to suit your specific enterprise needs. You may use the Radia System Explorer to enable the connections lists for the new UPGRADE class prior to adding the new instances.

Follow the steps below to have the UPGRADE class show up as a connectable class from the ZSERVICE class, and the PACKAGE class to be shown as a connectable class from the UPGRADE class through the **Show Connections** feature of the Radia System Explorer.

To add the Upgrade class to the Master Connection Rules

1. Use the Radia System Explorer and go to the PRIMARY.ADMIN.ZCONNECT class.
2. Right click Connection Rules (ZCONNECT) and select Edit class.

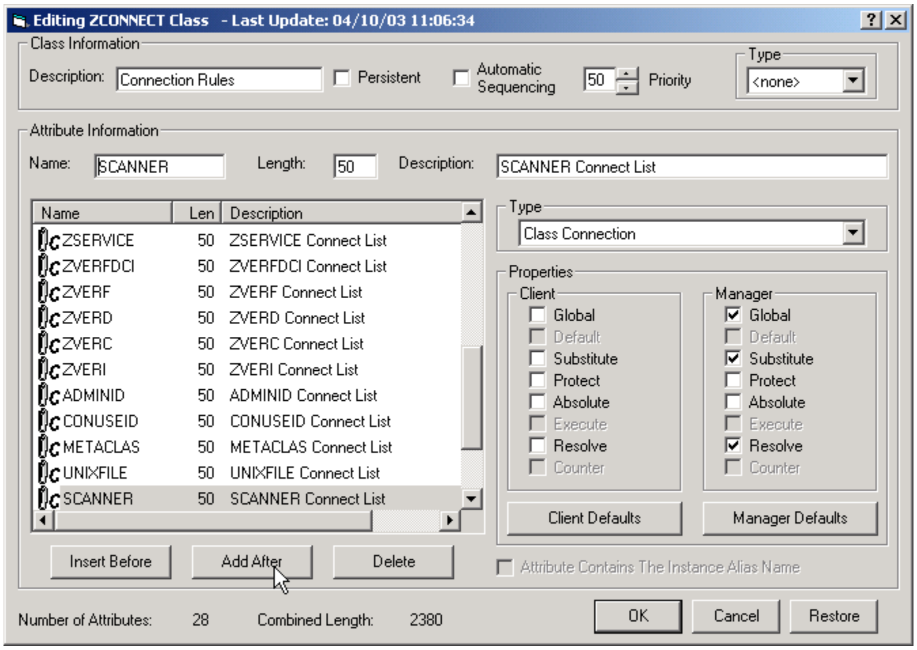


Connection Rules (ZCONNECT) Class Instances:

Name	Instance Name	Type
_BASE_INSTANCE_	_BASE_INSTANCE_	ADMIN.ZCONNECT Instance
_NULL_INSTANCE_	_NULL_INSTANCE_	ADMIN.ZCONNECT Instance
MASTER_CONNECTION_...	MASTER_CONNECTION_RU...	ADMIN.ZCONNECT Instance

PRIMARY\ADMIN\Connection Rules (ZCONNECT)\ 9/16/2003 9:13 AM

3 In the Editing Class window, scroll to the bottom of the class list, highlight the last template entry and click **Add After**.



Class Information

Description: Connection Rules ☐ Persistent ☐ Automatic Sequencing 50 Priority Type <none>

Attribute Information

Name: SCANNER Length: 50 Description: SCANNER Connect List

Name	Len	Description
ZSERVICE	50	ZSERVICE Connect List
ZVERFDCI	50	ZVERFDCI Connect List
ZVERF	50	ZVERF Connect List
ZVERD	50	ZVERD Connect List
ZVERC	50	ZVERC Connect List
ZVERI	50	ZVERI Connect List
ADMINID	50	ADMINID Connect List
CONUSEID	50	CONUSEID Connect List
METACLAS	50	METACLAS Connect List
UNIXFILE	50	UNIXFILE Connect List
SCANNER	50	SCANNER Connect List

Type: Class Connection

Properties

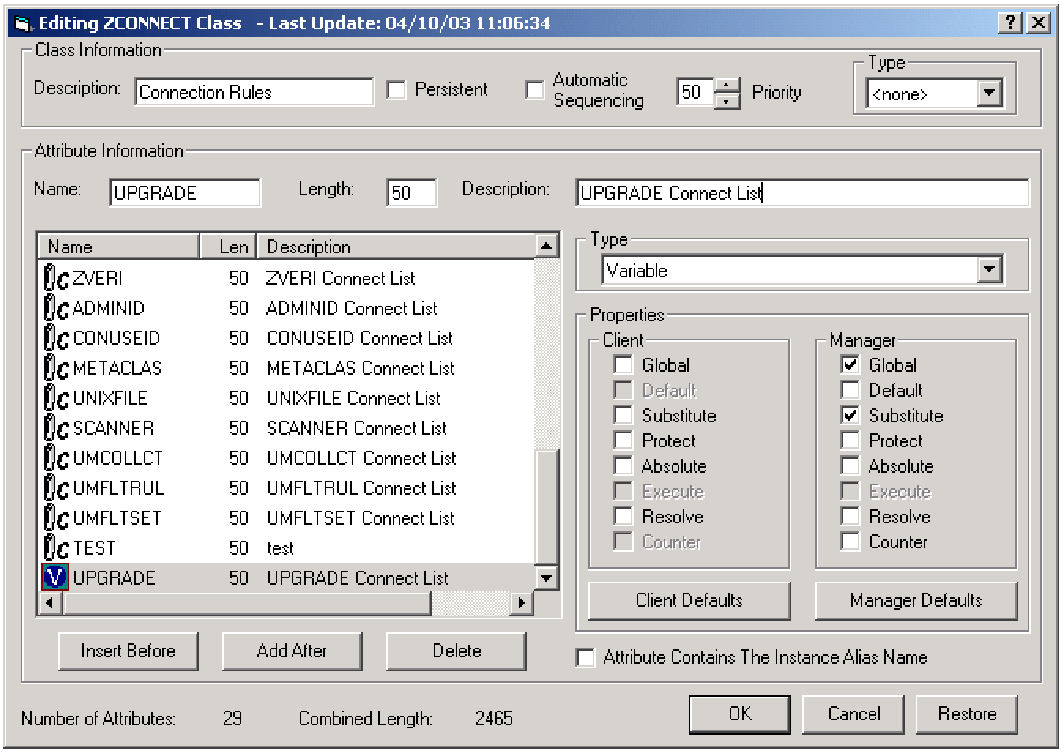
Client	Manager
☐ Global	☑ Global
☐ Default	☐ Default
☐ Substitute	☑ Substitute
☐ Protect	☐ Protect
☐ Absolute	☐ Absolute
☐ Execute	☐ Execute
☐ Resolve	☑ Resolve
☐ Counter	☐ Counter

☐ Attribute Contains The Instance Alias Name

Number of Attributes: 28 Combined Length: 2380

4 Add a new class connection field similar to the existing connection fields.

5 Name the new connection field UPGRADE and assign it a length of 50 and a description of UPGRADE Connect List.



Editing ZCONNECT Class - Last Update: 04/10/03 11:06:34

Class Information

Description: Connection Rules □ Persistent □ Automatic Sequencing 50 Priority

Type <none>

Attribute Information

Name: UPGRADE Length: 50 Description: UPGRADE Connect List

Name	Len	Description
ZVERI	50	ZVERI Connect List
ADMINID	50	ADMINID Connect List
CONUSEID	50	CONUSEID Connect List
METACLAS	50	METACLAS Connect List
UNIXFILE	50	UNIXFILE Connect List
SCANNER	50	SCANNER Connect List
UMCOLLCT	50	UMCOLLCT Connect List
UMFLTRUL	50	UMFLTRUL Connect List
UMFLTSET	50	UMFLTSET Connect List
TEST	50	test
UPGRADE	50	UPGRADE Connect List

Type: Variable

Properties

Client	Manager
□ Global	☑ Global
□ Default	□ Default
□ Substitute	☑ Substitute
□ Protect	□ Protect
□ Absolute	□ Absolute
□ Execute	□ Execute
□ Resolve	□ Resolve
□ Counter	□ Counter

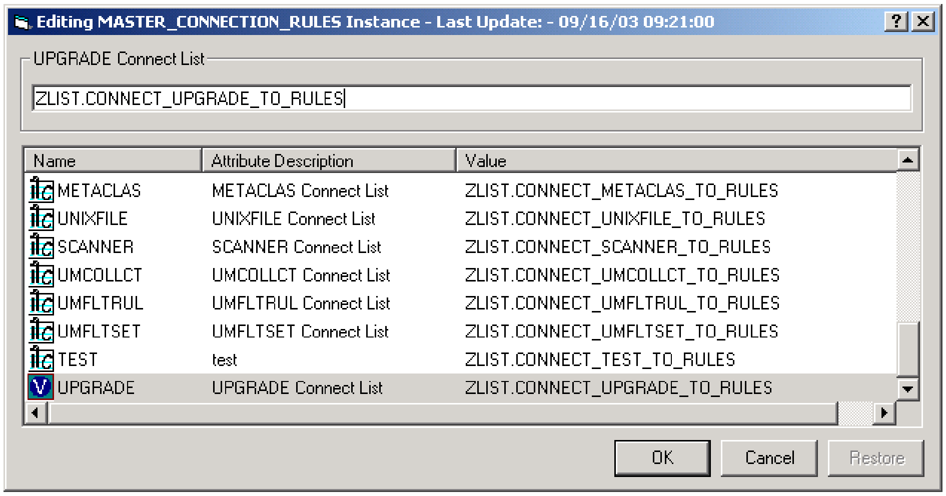
□ Attribute Contains The Instance Alias Name

Number of Attributes: 29 Combined Length: 2465

6 Click **OK** then **Yes.**

7 Right-click the PRIMARY.ADMIN.ZCONNECT.MASTER_CONNECTION_RULES instance and select **Edit Instance.**

8 For the new UPGRADE attribute, add the value ZLIST.CONNECT_UPGRADE_TO_RULES.



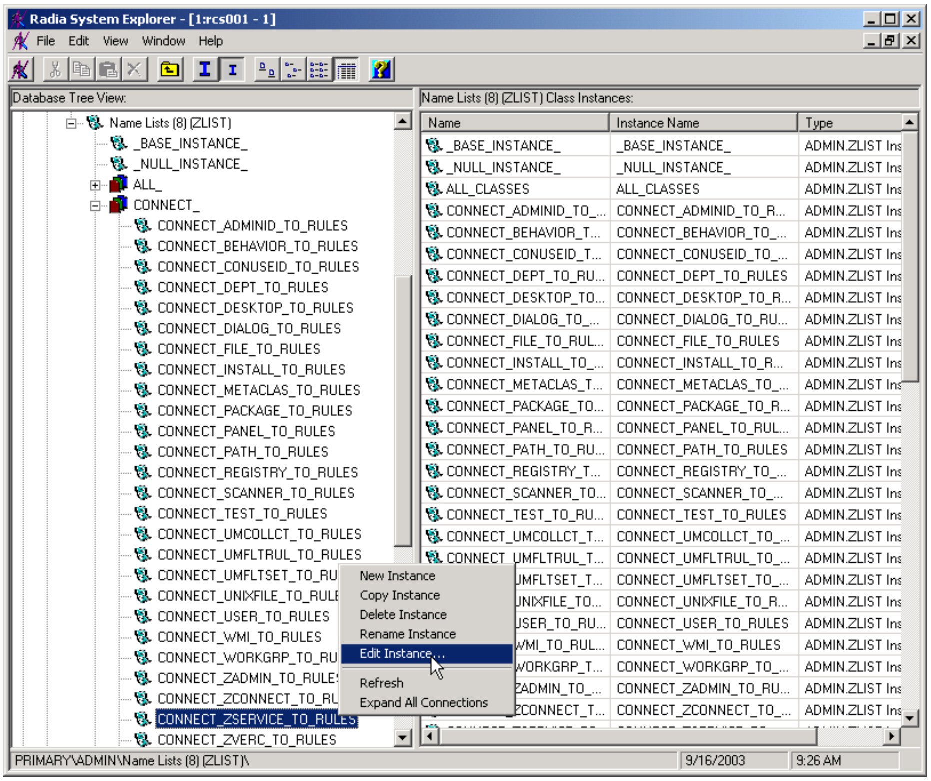
Editing MASTER_CONNECTION_RULES Instance - Last Update: - 09/16/03 09:21:00

UPGRADE Connect List

ZLIST.CONNECT_UPGRADE_TO_RULES

Name	Attribute Description	Value
METACLAS	METACLAS Connect List	ZLIST.CONNECT_METACLAS_TO_RULES
UNIXFILE	UNIXFILE Connect List	ZLIST.CONNECT_UNIXFILE_TO_RULES
SCANNER	SCANNER Connect List	ZLIST.CONNECT_SCANNER_TO_RULES
UMCOLLCT	UMCOLLCT Connect List	ZLIST.CONNECT_UMCOLLCT_TO_RULES
UMFLTRUL	UMFLTRUL Connect List	ZLIST.CONNECT_UMFLTRUL_TO_RULES
UMFLTSET	UMFLTSET Connect List	ZLIST.CONNECT_UMFLTSET_TO_RULES
TEST	test	ZLIST.CONNECT_TEST_TO_RULES
UPGRADE	UPGRADE Connect List	ZLIST.CONNECT_UPGRADE_TO_RULES

9 Right-click the PRIMARY.ADMIN.ZLIST.CONNECT_ZSERVICE_TO_RULES instance and select **Edit Instance**



Database Tree View:

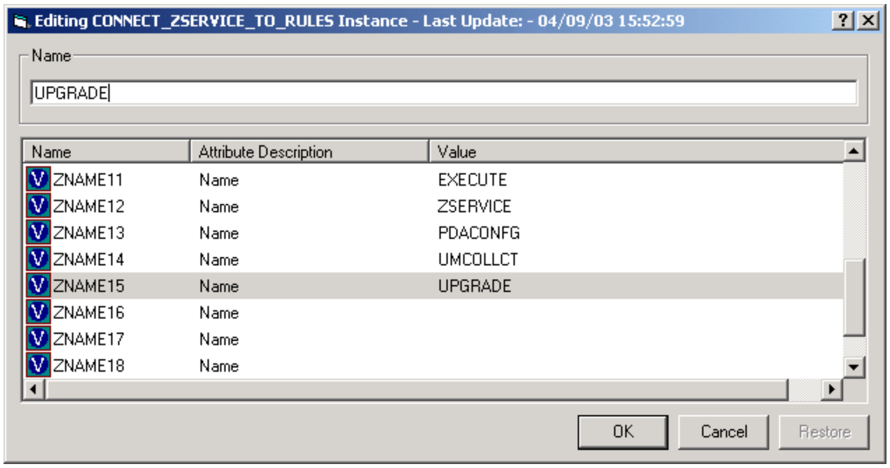
- Name Lists (8) (ZLIST)
  - _BASE_INSTANCE_
  - _NULL_INSTANCE_
  - ALL_
  - CONNECT_
    - CONNECT_ADMINID_TO_RULES
    - CONNECT_BEHAVIOR_TO_RULES
    - CONNECT_CONUSEID_TO_RULES
    - CONNECT_DEPT_TO_RULES
    - CONNECT_DESKTOP_TO_RULES
    - CONNECT_DIALOG_TO_RULES
    - CONNECT_FILE_TO_RULES
    - CONNECT_INSTALL_TO_RULES
    - CONNECT_METACLAS_TO_RULES
    - CONNECT_PACKAGE_TO_RULES
    - CONNECT_PANEL_TO_RULES
    - CONNECT_PATH_TO_RULES
    - CONNECT_REGISTRY_TO_RULES
    - CONNECT_SCANNER_TO_RULES
    - CONNECT_TEST_TO_RULES
    - CONNECT_UMCOLLCT_TO_RULES
    - CONNECT_UMFLTRUL_TO_RULES
    - CONNECT_UMFLTSET_TO_RU
    - CONNECT_UNIXFILE_TO_RULE
    - CONNECT_USER_TO_RULES
    - CONNECT_WMI_TO_RULES
    - CONNECT_WORKGRP_TO_RU
    - CONNECT_ZADMIN_TO_RULE
    - CONNECT_ZCONNECT_TO_RU
    - CONNECT_ZSERVICE_TO_RULES
    - CONNECT_ZVERC_TO_RULES

Name Lists (8) (ZLIST) Class Instances:

Name	Instance Name	Type
_BASE_INSTANCE_	_BASE_INSTANCE_	ADMIN.ZLIST Ins
_NULL_INSTANCE_	_NULL_INSTANCE_	ADMIN.ZLIST Ins
ALL_CLASSES	ALL_CLASSES	ADMIN.ZLIST Ins
CONNECT_ADMINID_TO_...	CONNECT_ADMINID_TO_R...	ADMIN.ZLIST Ins
CONNECT_BEHAVIOR_T...	CONNECT_BEHAVIOR_TO_...	ADMIN.ZLIST Ins
CONNECT_CONUSEID_T...	CONNECT_CONUSEID_TO_...	ADMIN.ZLIST Ins
CONNECT_DEPT_TO_RU...	CONNECT_DEPT_TO_RULES	ADMIN.ZLIST Ins
CONNECT_DESKTOP_TO...	CONNECT_DESKTOP_TO_R...	ADMIN.ZLIST Ins
CONNECT_DIALOG_TO_...	CONNECT_DIALOG_TO_RU...	ADMIN.ZLIST Ins
CONNECT_FILE_TO_RUL...	CONNECT_FILE_TO_RULES	ADMIN.ZLIST Ins
CONNECT_INSTALL_TO_...	CONNECT_INSTALL_TO_R...	ADMIN.ZLIST Ins
CONNECT_METACLAS_T...	CONNECT_METACLAS_TO_...	ADMIN.ZLIST Ins
CONNECT_PACKAGE_TO...	CONNECT_PACKAGE_TO_R...	ADMIN.ZLIST Ins
CONNECT_PANEL_TO_R...	CONNECT_PANEL_TO_RUL...	ADMIN.ZLIST Ins
CONNECT_PATH_TO_RU...	CONNECT_PATH_TO_RULES	ADMIN.ZLIST Ins
CONNECT_REGISTRY_T...	CONNECT_REGISTRY_TO_...	ADMIN.ZLIST Ins
CONNECT_SCANNER_TO...	CONNECT_SCANNER_TO_...	ADMIN.ZLIST Ins
CONNECT_TEST_TO_RU...	CONNECT_TEST_TO_RULES	ADMIN.ZLIST Ins
CONNECT_UMCOLLCT_T...	CONNECT_UMCOLLCT_TO_...	ADMIN.ZLIST Ins
CONNECT_UMFLTRUL_T...	CONNECT_UMFLTRUL_TO_...	ADMIN.ZLIST Ins
UMFLTSET_T...	CONNECT_UMFLTSET_TO_...	ADMIN.ZLIST Ins
UNIXFILE_TO...	CONNECT_UNIXFILE_TO_R...	ADMIN.ZLIST Ins
USER_TO_RU...	CONNECT_USER_TO_RULES	ADMIN.ZLIST Ins
WMI_TO_RUL...	CONNECT_WMI_TO_RULES	ADMIN.ZLIST Ins
WORKGRP_T...	CONNECT_WORKGRP_TO_...	ADMIN.ZLIST Ins
ZADMIN_TO_...	CONNECT_ZADMIN_TO_RU...	ADMIN.ZLIST Ins
ZCONNECT_T...	CONNECT_ZCONNECT_TO_...	ADMIN.ZLIST Ins

PRIMARY\ADMIN\Name Lists (8) (ZLIST)\ | 9/16/2003 | 9:26 AM

10 Add the value UPGRADE in the first empty variable slot.


Editing CONNECT_ZSERVICE_TO_RULES Instance - Last Update: - 04/09/03 15:52:59
OK
Cancel
Restore

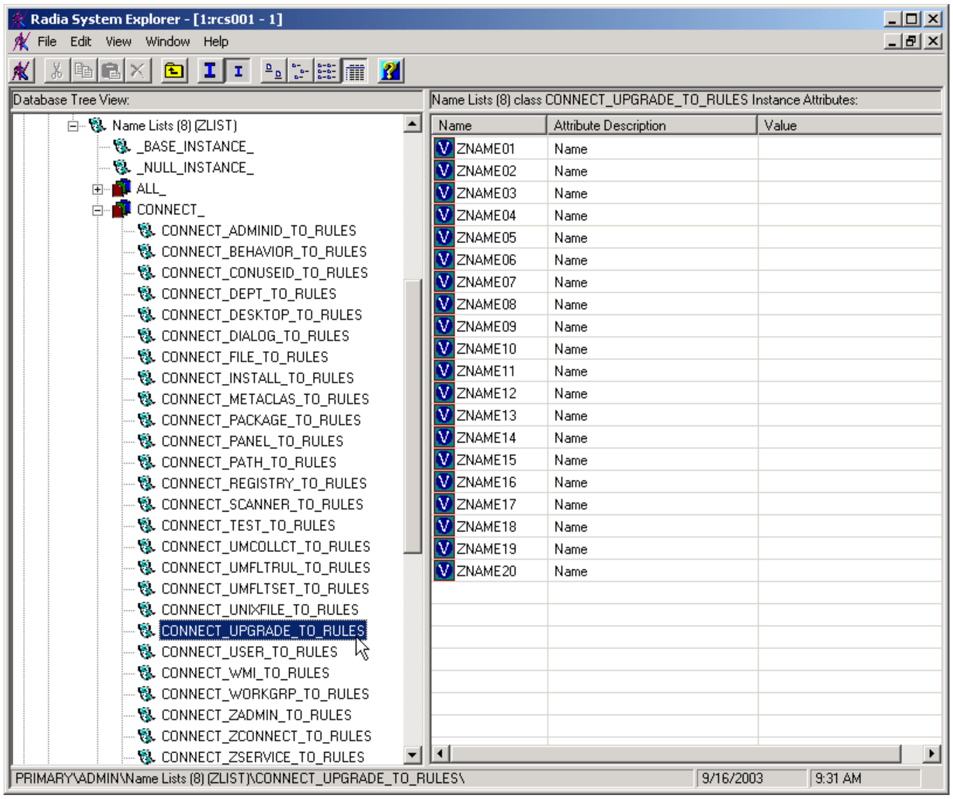
Name: UPGRADE

Name	Attribute Description	Value
ZNAME11	Name	EXECUTE
ZNAME12	Name	ZSERVICE
ZNAME13	Name	PDACONFG
ZNAME14	Name	UMCOLLCT
ZNAME15	Name	UPGRADE
ZNAME16	Name	
ZNAME17	Name	
ZNAME18	Name	

11 Click **OK.**

12 Add a new instance named to the ZLIST class by right clicking **Name Lists (8) (ZLIST)** and selecting **New Instance.**

13 Name the new instance **CONNECT_UPGRADE_TO_RULES**.



Database Tree View:

- Name Lists (8) (ZLIST)
  - _BASE_INSTANCE_
  - _NULL_INSTANCE_
  - ALL_
  - CONNECT_
    - CONNECT_ADMINID_TO_RULES
    - CONNECT_BEHAVIOR_TO_RULES
    - CONNECT_CONUSEID_TO_RULES
    - CONNECT_DEPT_TO_RULES
    - CONNECT_DESKTOP_TO_RULES
    - CONNECT_DIALOG_TO_RULES
    - CONNECT_FILE_TO_RULES
    - CONNECT_INSTALL_TO_RULES
    - CONNECT_METACLAS_TO_RULES
    - CONNECT_PACKAGE_TO_RULES
    - CONNECT_PANEL_TO_RULES
    - CONNECT_PATH_TO_RULES
    - CONNECT_REGISTRY_TO_RULES
    - CONNECT_SCANNER_TO_RULES
    - CONNECT_TEST_TO_RULES
    - CONNECT_UMCOLLCT_TO_RULES
    - CONNECT_UMFLTRUL_TO_RULES
    - CONNECT_UMFLTSET_TO_RULES
    - CONNECT_UNIXFILE_TO_RULES
    - CONNECT_UPGRADE_TO_RULES
    - CONNECT_USER_TO_RULES
    - CONNECT_WMI_TO_RULES
    - CONNECT_WORKGRP_TO_RULES
    - CONNECT_ZADMIN_TO_RULES
    - CONNECT_ZCONNECT_TO_RULES
    - CONNECT_ZSERVICE_TO_RULES

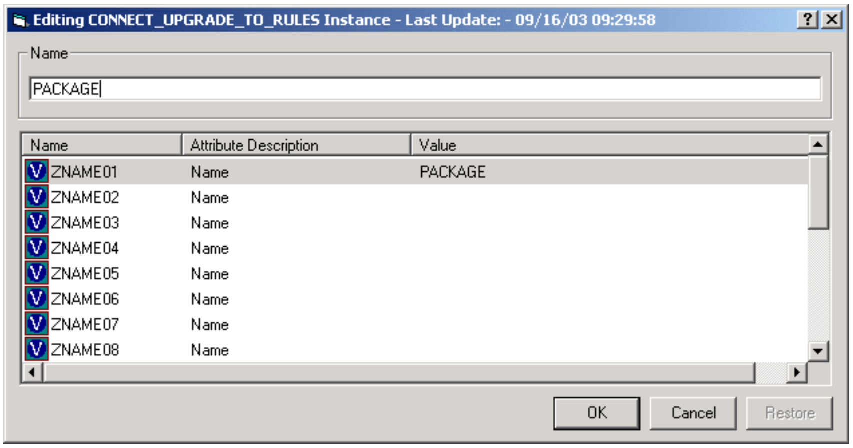
Name Lists (8) class CONNECT_UPGRADE_TO_RULES Instance Attributes:

Name	Attribute Description	Value
V ZNAME01	Name	
V ZNAME02	Name	
V ZNAME03	Name	
V ZNAME04	Name	
V ZNAME05	Name	
V ZNAME06	Name	
V ZNAME07	Name	
V ZNAME08	Name	
V ZNAME09	Name	
V ZNAME10	Name	
V ZNAME11	Name	
V ZNAME12	Name	
V ZNAME13	Name	
V ZNAME14	Name	
V ZNAME15	Name	
V ZNAME16	Name	
V ZNAME17	Name	
V ZNAME18	Name	
V ZNAME19	Name	
V ZNAME20	Name	

PRIMARY\ADMIN\Name Lists (8) (ZLIST)\CONNECT_UPGRADE_TO_RULES\ 9/16/2003 9:31 AM

14 Right-click the new instance and select **Edit Instance.**

15 Add the value **PACKAGE** to the first empty variable entry.



Editing CONNECT_UPGRADE_TO_RULES Instance - Last Update: - 09/16/03 09:29:58

Name: PACKAGE

Name	Attribute Description	Value
ZNAME01	Name	PACKAGE
ZNAME02	Name	
ZNAME03	Name	
ZNAME04	Name	
ZNAME05	Name	
ZNAME06	Name	
ZNAME07	Name	
ZNAME08	Name	

16 Click **OK.**

Completion of these steps shall enable the Radia System Explorer to display the UPGRADE class as a connectable class from the ZSERVICE class and the PACKAGE class to be shown as a connectable class from the UPGRADE class.

Mechanics of the Radia Client Upgrade for Unix when applied as a Radia Service

Once a user or machine's policy entitlement model is configured to include the Unix Client Upgrade materials, and the Radia Client connects and receives one of the mandatory application services XXX_41_RADIA_CLIENT_UPGRADE (where XXX represents the target operating system) containing the Radia Client Upgrade materials, Radia will download all of the necessary materials to perform a Radia Client Upgrade.

The default location of the upgrade media, when downloaded via a Radia client connect, is &(ZMASTER.ZSYSDIR)/ClientUpgrade (the location, as defined in UPGRADE.MEDIADIR attribute) in the machine's entitlement model. The installation of the Upgrade Service runs a method called "upgrade" (with additional command line parameters) which waits for Radia client processes to finish before upgrading the client binaries.

When the Radia Client processes have finished their execution, the actual client upgrade begins. The upgrade process stops the Radia client daemons (radsched and radexecd, if they are found to be running), upgrades the Radia client components specified by the **–installcomp** parameter of the UPGRADE class instance or the SelectComponent section of the specified configuration file, and restarts the Radia client daemons that were running at the time the upgrade was initiated.

After the upgrade service is distributed to a Radia client and the version 3.x upgrade process is complete, when the Radia client connects to the Radia Configuration Server again, the upgrade service will no longer be available to the user/machine. A ZSTOP expression exists in the upgrade instance, which evaluates to TRUE after the client is successfully upgraded to version 3.x. Once this expression evaluates to TRUE, the temporary service folder &(ZMASTER.ZSYSDIR)/ClientUpgrade and the underlying client upgrade media are removed from the client desktop.

Troubleshooting the Radia Client Upgrade for Unix

The Radia client upgrade creates an **upgrade.log** file in your log directory (IDMLOG). This file contains information regarding the upgrade and whether or not a successful upgrade has taken place. Review this file when the upgrade is complete, if your upgrade is not successful and more information is required, you can append **–loglevel 9** to the upgrade command line. This will set maximum logging for the client installer. Then run the client upgrade again.

The client installer log file **setup.log** produced by the increased log level can be found in the ~/tmp/setup directory.

Should you encounter a problem when running the Radia Client upgrade, provide be prepared to provide the files **upgrade.log** as well as **setup.log** files to Novadigm Technical Support.

Upgrading the Radia Client to Version 3.x – from CDROM

To upgrade the Client – Local upgrade method

1. Login to the Unix workstation as root
2. From the **Radia 4 Upgrade CD**, go to **/client/***platform* (where platform denotes the operating system on which you are performing the client upgrade)……….

   Then type:

   ```
   ./upgrade –c "–installcomp <PARMS> –mgrip<IP> –mgrport <PORT>"
   ```

   Where:

- The text PARMS is either a single argument or a comma separated list of Radia client components that you wish to upgrade.

Valid Client components include:

ram	- specifies Radia Application Manager
rim	- specifies Radia Inventory Manager
rsm	- specifies Radia Software Manager

- The text IP would be replaced with the literal IP address of the Radia Configuration Server from which you would like your upgraded Radia client to receive new applications.
- The text PORT would be replaced with the port number on which the Radia Configuration Server is running.

The **upgrade** command line is used to specify the command line arguments to be passed to the client upgrade process. Because all arguments specified after the –c option are passed directly to the client installer, any arguments that can be specified on the client install command line can be specified here except for the –mode argument which is automatically set by the upgrade script to perform a silent install/upgrade.

The command line arguments following the –c option must be placed inside double quotes.

The **–installcomp** command line option is used to specify the specific products to be upgraded or installed on the client workstation. For example, to upgrade your Radia Application Manager and also install the Radia Inventory Manager and Radia Software Manager, you would specify "-installcomp ram,rim,rsm".

When multiple products are specified in the command line arguments the Radia product acronyms must be comma separated.

The **–mgrip** and –**mgrport** command line options specify the Radia Configuration Server ip address and port that the upgraded client will use for it's ZMASTER object and sample shell scripts. Should you upgrade or install the Radia Software Manager the Software Manager's configuration file called args.xml would be populated with Radia Configuration Server IP address and port number specified with the **–**

mgrip and **–mgrport** command line options. The default value for –mgrport is 3464.

You can specify **–loglevel 9** in the command line to set maximum logging for the client upgrade/install.

Alternatively, you can perform a local client upgrade using a Client Install configuration file.

To perform a local client upgrade using a Client Install configuration file

1. Login to the Unix workstation as root.
2. From the **Radia 4 Upgrade CD**, go to **/client/***platform* (where platform denotes the operating system on which you are performing the client upgrade)……….

   Should you have an installation configuration file already on the Client workstation, (created by the Remote Installation Setup process of the Radia client installer), you can have the client upgrade use this configuration file as a basis to perform a Client upgrade by typing:

   ```
   ./upgrade –c "–cfg configfilename.cfg"
   ```

   Where configfilename.cfg specifies a fully qualified path name to an install configuration file that contains the components selection and desired configuration options.

   For information about installer configuration files (created by the Remote Installation Setup process of the Radia client installer), please consult the Client installation chapter of the Radia Application Manager, Radia Inventory Manager, and/or the Radia Software Manager Guides for Unix.

It is not recommended to specify both a configuration file and command line options **–installcomp**, **–mgrport** or **–mgrip** because the command line options will override the values specified in the configuration file.

For example, the command line of –c " –loglevel 9 –cfg /tmp/myown.cfg" would rely exclusively on the variables set in the /tmp/myown.cfg configuration file to determine what Radia components to upgrade/install and what Radia Configuration Server IP and port to use as well as other upgrade/install options.