RoamServer 5 for Unix Administrator Guide

v1.1

Contents

For the latest version of this document, please check the iPass FTP site. This is version 1.1.

RoamServer 5 for Unix Release Notes

I. Introduction

The **RoamServer 5 for Unix Administrator Guide** provides step-by-step instructions for the installation and configuration of RoamServer 5 for Unix. It also includes instructions on how to configure RoamServer to use Unix, Site, RADIUS, LDAP or TACACS+ as an authentication protocol.

You should read this guide all the way through before attempting to install, upgrade, or configure RoamServer software.

For the latest information on RoamServer 5, check the RoamServer 5 for Unix Release Notes.

Note: these instructions refer to a directory called <iPass_Home>. This is the directory where your RoamServer is installed; the default is /usr/ipass/roamserver. (Note that in previous versions of RoamServer, the default was /usr/ipass.)

System Requirements

Minimum Server Requirements

The iPass RoamServer software for Unix platforms must be installed on a server that meets the following requirements:

- 110 MB temporary disk space
- 70 MB permanent disk space
- Root access is required for installation
- Server must have a static IP address (no DHCP)
- If installed behind a firewall, an accessible NAT IP address
- Installer must have administrative permissions on the machine
- Solaris users need the Solaris SPARC OS Patches required for the Java 2 Standard Edition (J2SE) 1.4 from http://java.sun.com/j2se/1.4/download.html .(If the web page has moved, go to http://java.sun.com/ to find the download page.)
- A Linux host needs at least 20 threads of 30 MB each and 600 MB swap space.

Additional Requirements

- Connectivity to an authentication database
- The TCP/IP protocol is required to support the SSL encrypted connection from the iPass Transaction Centers
- If placed behind a firewall, the firewall must not block inbound connections to TCP port 577, or any other agreed-upon port. The firewall must not block outbound connections to the iPass Transaction Centers. See Installing Behind a Firewall for more details.

Preferences

In addition it is preferable, although not required, for your server to have:

- Connectivity to an accounting server to allow accounting logs to be written to an alternate location

Available Platforms

RoamServer 5 has been successfully tested on the following Unix platforms:

- Solaris 7, 8, and 9 (SPARC architecture processor)
- Linux RedHat 6.1 (Kernel 2.2.12) and 6.2 (Kernel 2.4.14) with the Workstation Package

installed. This package includes libraries required by the installer and the server.

II. Installation of RoamServer 5

Before You Begin

Before beginning, you will need the following:

- Administrator rights on the RoamServer host.
- Your iPass Code, given to your company when it signed up with iPass. Please contact your Installation Engineer if you don't have this code.

In addition, please be prepared to supply the following to your iPass Installation Engineer:

- Your host's IP address
- Your host's public IP address
- The port number on which RoamServer will listen
- The host's operating system, including kernel and version number.

Installation Outline

The installation process consists of the following steps:

1. **Download the installation file. (For a Solaris host, see Solaris Installation, below.)**
2. **Install the software.**
3. **Set initial configuration and certify the RoamServer.**
4. **Configure RoamServer to communicate with your authentication server(s), and if desired, accounting server(s).**
5. **Set any advanced options, such as:**
   - **Policy File**
   - **Secondary Servers for Failover**
   - **Log Files**
6. **Set additional properties in the ipassRS.properties file, if necessary, including Ascend Data Filters for non-VPN access.**
7. **Test the installation.**

Solaris Installation

If installing on a Solaris host, you must first download the latest kernel patches from the Sun Web site for Java 1.4.0 Please see http://java.sun.com for more information.

Installing Behind a Firewall (Recommended)

iPass recommends that you install RoamServer behind a firewall. If you choose to follow this recommendation, you will need to allow TCP traffic to the external IP of the RoamServer on port 577 through to the RoamServer. In addition, iPass will need a valid public IP address to put into its database. You may restrict traffic on that port to only allow incoming packets from the following IP addresses:

Amsterdam, NL: 216.239.102.125	San Paulo, BZ: 216.239.101.125
Atlanta, US: 216.239.111.125	Santa Clara, US: 216.239.99.125
Frankfurt, DE: 216.239.108.125	Sydney, AU : 216.239.98.125
Hong Kong, HK: 216.239.110.125	Tokyo, JP: 216.239.109.125
London, UK: 216.239.105.125	TBD: 216.239.103.125
New York, US: 216.239.104.125	TBD: 216.239.107.125
Palo Alto, US: 208.212.202.21	

However, you may be asked to open the port to other IPs as the iPass network continues to grow. The most current list of IP addresses is posted on the iPass secure Web site.

Note: If your firewall is performing Network Address Translation (NAT), you will need to provide the IP address of your firewall to your iPass Installation Engineer.

Downloading

Before installing, you will need to download the installation file from the iPass FTP site.

To download the installation file using FTP:

1. FTP to ftp.ipass.com
2. Enter your user name and password given to you by your iPass installation engineer.
3. To change to binary mode, type: bin
4. To obtain a complete listing of directory contents, type: dir
5. To change to the specific directory containing the software specific to your platform and the region you are in, type: CD. Remember that directory names and filenames are case-sensitive.
6. After locating the file appropriate to your platform and region, type: get rs5setup.bin.
7. To exit the ftp application, type: bye
8. We recommend you read all enclosed documentation before continuing.

Installing the Software

You must uncompress and untar the installation files to install the RoamServer software.

To install the RoamServer 5 directories:

1. As root, type chmod +x rs5setup.bin
2. Type rs5setup.bin to run the installation program.

This will create a hierarchy in /usr/ipass/roamserver with all the necessary directories and files.

Note: In order for RoamServer to run correctly, you *must* keep the file structure as it is installed. However, RoamServer can be installed in any location.

Upgrading to RoamServer 5

If you've upgraded to RoamServer 5 from a previous version, after RS 5 is successfully installed, the RS 3.7.2 entry in the /etc/services file must be removed before starting up the new RS 5.0

Page Top

III. Initial Configuration and Certification

Before first running RoamServer 5, you must perform some initial setup tasks and receive a digital certificate from iPass. This section explains how to complete these tasks.

The ipassRS.properties File

The main RoamServer configuration file is called *ipassRS.properties* and it controls customization of your RoamServer functionality. By setting properties in the file, you can enable or disable RoamServer functions. Some properties are turned on by default, and it is necessary to change the value of the property in order to turn off that functionality. (Enabling some functionality involves setting more than one attribute.)

You can edit the file and add, change or delete properties in one of two ways:

- You can run ipassconf.csh in your <iPass_Home>/bin directory. This is the recommended method and is explained in detail below.
- You can also use a text editor. To set a new property value in a text editor, simply open the file and type in the name and value of a new attribute. (However, we **strongly recommend** use of the ipassconf.csh script, when possible, to ensure correct naming and formatting of property names and values.)
    - If a text editor is used, properties should be set as follows:

      <property name>=<value>.

    Property names are case-sensitive, but property values are not. Valid values for boolean properties are: true, false, yes, no, y, n.

Running *ipassconf.csh*

Configuration tasks can be performed quickly and easily by running a script called ipassconf.csh, located in the <iPass_Home>/bin directory, which can be used to set properties in the ipassRS.properties file.

To run *ipassconf.csh*:

1. in your <iPass_Home>/bin directory, type ipassconf.csh.

2. The script requests important configuration information. Enter the requested information as needed.

3. The values in square brackets [ ] are default values. To enter a default value, press Enter.

Multiple instances of running ipassconf.csh are not recommended. You should only run a single instance of the script at any one time, as simultaneous instances can overwrite each other's results.

Adding, Editing or Deleting Properties

You can rerun the script after initial configuration to add, edit or delete properties, as needed. If you rerun it, the script will read the default values from the existing iPassRS.properties, so you won't have to re-enter those values.

For example, two months after you install the RoamServer software, you decide to add a secondary authorization server. Simply run the ipassconf.csh, skip all the questions not having to do with authorization servers by entering default values (press Enter each time), and only enter the configuration information for the new authorization server when the script requests this information.

Initial Configuration

Initial configuration is done by running the ipassconf.csh script, which sets many of the properties in your *ipassRS.properties* file.

To initially configure your RoamServer:

1. In the<iPass_Home>/bin directory, run *ipassconf.csh*. Supply the requested information as outlined here. For each script entry, the value shown in square brackets [ ] is the default .Where applicable, you can press Enter to use default values for the information.

2. **Time and Date Verification:** (Default Value=YES.) The date/time stamp must be correct and correspond with the information in the iPass database in order to validate the certificate.

3. **Customer ID:** (Default Value=N/A) Enter your customer ID, supplied by iPass. This is the same ID number used on your iPass Web site login.

4. **Policy File:** (Default Value=No) If you want to use a Policy File to allow or deny users access, enter Yes.

5. **Debug Level:** (Default Value=0):Debug level determines how debugging and error messages are logged to a trace file. Debug level can be any value from 0 to 5, with 0 generating only critical error messages and 5 generating the most detailed and extensive amount of

information. Production servers should normally be run with a debug level of 0.

6. **Operating System:** (Default Value=OS). Enter your operating system as appropriate. Please ensure that the operating system is entered correctly for your host, which should be either SunOS or Linux, as appropriate.

7. **Port:** (Default Value=577) Enter the RoamServer listening port. iPass recommends you use port 577.

8. **Authentication Servers:** (Default Value=no). If you wish to configure your authentication server(s), enter yes. You will need to enter each server's authentication protocol, IP address and other relevant configuration parameters. See Authentication Servers for more information.

9. **Accounting Servers:** (Default Value=no). If you wish to configure your accounting server(s), enter yes. You will need to enter each server's IP address and other relevant configuration parameters. Note that this is not mandatory since iPass records all accounting information at in its central settlement system. See Accounting Servers , for more information.

10. **SSL Certificate: Enter the information needed to generate your SSL certificate, including:**
    - **2-character Country Code:** (Default Value=US)
    - **State or Province Name:** (Default Value=Some-State)
    - **City or Town Name:** (Default Value=Some-City)
    - **Company or Organization Name:** (Default Value=Some-Organization)
    - **Public IP Address of the RoamServer Host:** (Default Value=Local IP). This must be the public or external IP address, and may differ from the IP address you entered above. The IP address will not be stored by RoamServer but will be used to generate your public key certificate. If you are using NAT (Network Address Translation), please supply this external address to your iPass installation engineer as well.
    - **Fully Qualified Domain Name of the RoamServer Host:** (Default Value=N/A). The domain name will not be stored by RoamServer but will be used to generate your public key certificate.
    - **Your E-mail Address:** (Default Value=N/A). iPass will e-mail your certificate to this address after processing. iPass recommends that this mailbox be accessible to the host on which you are installing the software. (You will need to be able to transfer this certificate to the RoamServer host.)

Processing: The script will then describe any errors that may have occurred during installation and generate a certificate request, as follows:

- A copy of the certificate is saved as <iPass_Home>/certs/mail_cert_req.data.

- If SMTP services are available on this server, you should mail it to *cert-request@ipass.com* as an attachment.

- If SMTP services are *not* available on this server, the preferred method of certificate request exchange is in real time with an iPass technician using FTP. Contact your iPass installation engineer to arrange this exchange.

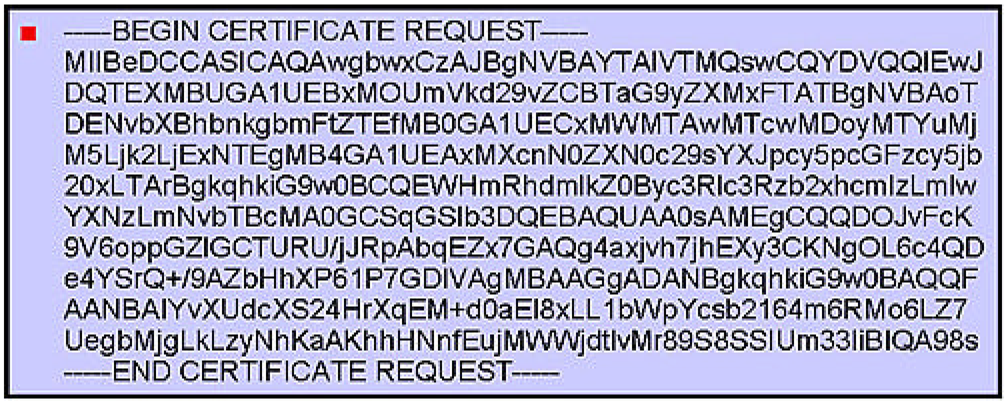
Certification

Upon successful completion of the certificate request generation script, iPass will process your request and generate a certificate for your RoamServer, which will be valid for 10 years. The x509 certificate will allow SSL 128-bit encrypted communication between the iPass transaction server and your RoamServer.

NOTE: *This may take between 1 hour to 2 days. If you need the certificate immediately, please contact iPass Technical Support.*

Once your certificate request is processed, iPass will send the certificate file back to you, either via e-mail or FTP. You will need to save the information in this file, without alteration, as a file named *isp_cert.pem* in the *<iPass_Home>/certs/* directory.

If you are cutting and pasting the file from an e-mail, be sure to include the header and footer of the certificate string as shown in the example certificate below.



```
-----BEGIN CERTIFICATE REQUEST-----
MIIBeDCCASICAQAwgbwxCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJ
DQTEXMBUGA1UEBxMOUmVkd29vZCBTaG9yZXMxFTATBgNVBAoT
DENvbXBhbnkgbmFtZTEfMB0GA1UECxMWMTAwMTcwMDoyMTYuMj
M5Ljk2LjExNTEgMB4GA1UEAxMXcnN0ZXN0c29sYXJpcy5pcGFzcy5jb
20xLTArBgkqhkiG9w0BCQEWHmRhdmlkZ0Byc3Rlc3Rzb2xhcmlzLmlw
YXNzLmNvbTBcMA0GCSqGSIb3DQEBAQUAA0sAMEgCQQDOJvFcK
9V6oppGZIGCTURU/jJRpAbqEZx7GAQg4axjvh7jhEXy3CKNgOL6c4QD
e4YSrQ+/9AZbHhXP61P7GDIVAgMBAAGgADANBgkqhkiG9w0BAQQF
AANBAIYvXUdcXS24HrXqEM+d0aEI8xLL1bWpYcsb2164m6RMo6LZ7
UegbMjgLkLzyNhKaAKhhHNnfEujMWWjdtlvMr89S8SSIUm33IiBIQA98s
-----END CERTIFICATE REQUEST-----
```

Verification

You have now completed the basic installation process, and are almost ready to begin testing. Before you begin, you will need to verify and configure the main configuration file for the Roamserver software, named *<iPass_Home>/ipassRS.properties*. This can be opened in a text editor. Examine this file and verify that all values have been entered correctly. See The ipassRS.properties File.

Run the script *verify_certificate* in *<iPass_Home>/bin* to verify the installed certificates.

If all of the above has been completed successfully, your Roamserver is now ready for testing. For test information and instructions, see Testing.

Setting Your RoamServer to Automatically Restart

We recommend that you configure your RoamServer to automatically restart, in case your RoamServer host cycles power or reboots. You can do this one of two ways: manually, or, if your system is at runlevel 3, by a script.

Find the runlevel of your system by typing the following:

- In Solaris, type who -r

- In Linux, type runlevel

If your system is at runlevel 3, you can use a script to configure auto-restart.

To configure a runlevel 3 system automatically for auto-restart: in <iPass_Home>/scripts, run the script setupRoamserverBootScript.sh

Or, to configure auto-restart manually:

1. Copy the file roamserverd from <iPass_Home>/scripts to your local /etc/init.d directory.
2. Make sure roamserverd is executable.
3. Change directory to /etc/rc(n).d, where n=runlevel. For example, for a runlevel 3 system, change directory to /etc/rc3.
4. In rc(n).d, make a file named: S[nn]roamserverd, where nn=order that this file will be executed in, from 00 to 99. nn should be higher than any of the existing files in this directory. For example, S99roamserverd.
5. Softlink this new file to the script /etc/init.d/roamserverd by entering: <filename> ->/etc/init.d/roamserverd. For example, S99roamserverd->/etc/init.d/roamserverd
6. When the host restarts, the RoamServer will also restart.

Page Top

IV. Authentication and Accounting Servers

This section provides instructions for configuring your iPass RoamServer software to communicate with your authorization and accounting servers.

These instructions assume that you are installing the RoamServer software behind your firewall and/or on the same machine as your AAA server. If you are installing the RoamServer in front of your firewall or even on the firewall server, you may need to modify some of these settings. Consult with your iPass RoamServer Installation Engineer for assistance.

Unix and SITE Authentication

If you would like the RoamServer to authenticate using your Unix system's password file, the type of authentication protocol you choose will be based on the type of passwords used.

- If your system does not use shadow passwords, Unix authentication should be used.

To enable Unix authentication:

1. Run ipassconf.csh.
2. When the script requests authorization server information, enter Unix.

- If your system uses shadow passwords, SITE authentication should be used instead.

To enable SITE authentication:

1. Run ipassconf.csh.
2. When the script requests authorization server information, enter Site.
3. For Site File, enter /etc/shadow. (**Note:** on some systems, for instance Linux systems in particular, your password file may be /etc/master/shadow or some other file. Use whatever name is appropriate for your system.)

Page Top

RADIUS Authentication

The iPass RoamServer can forward authentication requests and accounting packets, if desired, to a RADIUS server running on the network. The RoamServer will format the request as a standard RADIUS request and forward it to the RADIUS daemon at the address and port number that is specified during the installation. You must know the IP address and port number that will be used to reach your RADIUS server. Additionally, you must make the RADIUS encryption key (shared secret) available to the RoamServer software. The RoamServer uses this shared secret to encrypt the RADIUS packet contents before sending them to the RADIUS server. The RADIUS server then uses the shared secret to decrypt the packet contents.

Note: Your system must have a static, routable IP address, and cannot be blocked by a firewall.

To configure your RoamServer for RADIUS authentication:

1. Run ipassconf.csh. Enter radius as an authentication protocol and enter:
   - the server's IP address [127.0.0.1]
   - port number [1812]
   - RADIUS shared secret [mysecret]
   - attempts [3]
   - idle timeout in milliseconds[5000]
   - if the prefix should be included [N]
   - if the domain should be included [N]

2. Verify that the RoamServer is entered as a client of your RADIUS. You will need to edit the appropriate configuration file within your RADIUS software by adding the IP address of the RoamServer, and the corresponding shared secret, that you entered in above.

3. If you make any changes to your RADIUS, you will have to restart it to make sure the changes take effect.

4. Restart your RoamServer. The RoamServer will now be able to authenticate against your RADIUS Server.

NOTE: The RoamServer can contain the IP address of more than one Authentication or Accounting Server for failover purposes. For more information, please see Setting Secondary Authentication or Accounting Servers for Failover for details.

Page Top

LDAP Authentication

The iPass RoamServer can forward authentication requests to an LDAP server running on the network. The RoamServer will format the request as a standard LDAP request and forward it to the LDAP daemon at the address and port number that is specified during the installation. You must know the IP address and port number that will be used to reach your LDAP server. Additionally, you must configure/customize how the RoamServer software will perform authentication at the LDAP server. LDAP specific configurations are set in a file called *ipassLDAP.properties*. For more information, please refer to the Appendix in this guide, and the *ipassLDAP.properties.example* file included in the RoamServer software package.

To configure RoamServer for LDAP authentication:

1. Open the file named *<iPass_Home>/ipassLDAP.properties*. If this file does not exist, please create it. For an illustration, see the file: ipassLDAP.properties.example.

2. Run ipassconf.csh. Enter LDAP as an authentication protocol and enter:

   - the server's IP address
   - LDAP configuration file name
   - port number [389]
   - number of attempts
   - idle timeout in milliseconds [10000]

3. Customize the contents of the *ipassLDAP.properties* file as needed. See The ipassLDAP.properties Configuration File for more details.

4. Save and exit the file.

5. Restart your RoamServer. The RoamServer will now be able to authenticate against your LDAP server.

NOTE: The RoamServer can contain the IP address of more than one Authentication or Accounting Server for failover purposes. For more information, please see Setting Secondary Authentication or Accounting Servers for Failover for details.

For more details about configuring your RoamServer for LDAP authentication, see Appendix: The ipassLDAP.properties Configuration File .

Page Top

TACACS+ Authentication

The iPass RoamServer can forward authentication requests to a TACACS+ server running on the network. The RoamServer will format the request as a standard TACACS+ request and forward it to the TACACS+ daemon at the address and port number that is configured during the installation. You must know the IP address and port number that will be used to reach your TACACS+ server. Additionally, you must make the TACACS+ shared secret available to the RoamServer software. The shared secret is configured in the TACACS+ configuration file called *tac_plus.conf*. The RoamServer uses this shared secret to encrypt the TACACS+ packet contents before sending them to the TACACS+ server. The TACACS+ server then uses the shared secret to decrypt the packet contents. Please refer to your TACACS+ documentation for more information on the *tac_plus.conf* file and shared secret. The TACACS+ server can be located anywhere with a routable, static IP address, including on the same machine as the RoamServer.

Note: If the TACACS+ server is running on an alternative machine on your network (i.e. not on the server running RoamServer), you will need to install a copy of the *tac_plus.conf* file on that server or on a network-addressable drive available to that server. You will also need to inform the RoamServer of the location of this file during configuration.

To configure the RoamServer for TACACS+ authentication:

1. Run ipassconf.csh. Enter tacacs+ as an authentication protocol and enter:
    - the server's IP address
    - port number
    - TACACS+ Shared Secret
    - number of attempts
    - idle timeout
2. Open *<iPass_Home>/tac_plus.conf.example*
3. Verify the settings in the configuration file for your TACACS+ server. You may need to edit the appropriate configuration file within your TACACS+ software by adding the IP address of the RoamServer.
4. If you make any changes to your TACACS+, you will have to restart it to make sure the changes take effect.
5. Restart your RoamServer. The RoamServer will now be able to authenticate against your TACACS+ server.

NOTE: The RoamServer can contain the IP address of more than one Authentication or Accounting Server for failover purposes. For more information, please see Setting Secondary Authentication or Accounting Servers for Failover for details.

Page Top

V. Options

Using a Policy File

A Policy File allows you to filter the requests being sent to your authentication server. The RoamServer will validate all users against this file before contacting your authentication server. This feature may be helpful if you wish the RoamServer to authenticate from a large user database, but only want a small group of those users to be able to roam, or conversely, if you only wish to deny roaming access to a small group.

The Policy Tool, rs_policy, located in your <iPass_Home>/bin directory, is an application used for creation and maintenance of your Policy File. Although the Policy File is a text file, iPass recommends you use the Policy Tool for creating, editing and maintaining your Policy File. This will ensure proper formatting and correct policy criteria.

To create a policy file, in the <ipass_Home>/bin directory, run the file rs_policy. If the tool detects that no Policy File exists, it will create one in the default directory.

To enable use of a Policy File,

1. Run ipassconf.csh.
2. At the line, *Do you wish to use the policyFile for authentication?*, enter Yes.
3. Enter the path and name of your policy file, or press Enter to accept the default.

To edit or manage your policy file:

1. Run the Policy Tool.
2. Choose your option from the options menu, and follow the tool prompts. You can
   - Add a rule
   - Remove a rule
   - Explain an existing rule
   - List the rules
   - Save the rules
   - List a country code
   - Quit

3. When done, enter 9 to quit the Policy Tool.

Policy File Mapping

This table shows the mappings of NAS port type numbers to the class of service.

nas-port-type	Class of Service
0	Dial-up
1	Dial-up
2	Dial-up ISDN
3	Dial-up ISDN
4	Dial-up ISDN
5	Dial-up
6	Dial-up PIAFS (PHS)
7	Dial-up
8	Dial-up
9	Dial-up
10	Dial-up
11	Wired
12	Wired
13	Wired
14	Wired
15	Wired
16	Wired
17	Wired
18	Wireless
19	Wireless

Page Top

Setting Secondary Authentication or Accounting Servers for Failover

If the primary server is unreachable, the iPass RoamServer can fail over to one or more secondary authentication/accounting servers. This feature works with RADIUS, LDAP and TACACS+ authentication protocols.

Your secondary servers do not have to be of the same type as your primary server. For instance, if you had both a RADIUS server and an LDAP server, you could designate your RADIUS server as primary and your LDAP server as secondary, or vice versa.

To configure the RoamServer to fail over to a secondary authentication server:

1. Run ipassconf.csh.

2. At the line, *Do you wish to add new Auth server?,* enter *Yes.*

3. Enter the properties for the new authorization server as described under Authorization Servers, above.

4. If you are using RADIUS or TACACS+, you must make sure that the shared secrets are the same for each server. Also, if using RADIUS, you must make sure that the RoamServer is entered as a client of the Secondary RADIUS as well as with the Primary.

5. Restart your RoamServer. The RoamServer should now be able to fail over to the secondary authentication server in the case of a power, hardware, or software failure.

Setting More Secondary Servers: There is no limit to the number of secondary authorization servers you can specify. You can repeat the above to specify more authorization servers, incrementing the number for each new server (AuthServer1, AuthServer2, etc.). However, in the ipassRS.properties file, you must ensure that servers are listed in numerical order such as : AuthServer1, AuthServer2, AuthServer3, or failover will not occur. Also, you may not skip any numbers in the sequence when specifying servers. (For example, AuthServer1, AuthServer2 and AuthServer4 would not an acceptable sequence.)

To configure the RoamServer to fail over to a secondary accounting server:

1. Run ipassconf.csh.

2. At the line, *Do you wish to add new Acct server?,* enter *Yes.*

3. Enter the properties for the new accounting server as described under Authorization Servers, above.

4. If you are using RADIUS or TACACS+, you must make sure that the shared secrets are the same for each server. Also, if using RADIUS, you must make sure that the RoamServer is entered as a client of the Secondary RADIUS as well as with the Primary.

5. Restart your RoamServer. The RoamServer should now be able to fail over to the secondary accounting server in the case of a power, hardware, or software failure.

Setting More Secondary Servers: There is no limit to the number of secondary accounting servers you can specify. You can repeat the above to specify more accounting servers, incrementing the number for each new server (AcctServer1, AcctServer2, etc.). However, in the ipassRS.properties file, you must ensure that servers are listed in numerical order such as : AcctServer1, AcctServer2, AcctServer3, or failover will not occur. Also, you may not skip any numbers in the sequence when specifying servers. (For example, AcctServer1, AcctServer2 and AcctServer4 would not an acceptable sequence.)

A Note About Failover and Local Servers

Since there will always be a response from the local server, if you set one of your failover servers to Unix or Site, there is no need to set any further servers in the sequence.

Page Top

Trace Log File Configuration

The RoamServer can be configured to write information about access attempts to a log file for debugging purposes. If enabled, debugging information is output to a local log file, named *roamserver.trace*, which is found in the *<iPass_Home>/logs* directory. The amount of debugging output can be controlled by changing the value of the *Debug Level* parameter. (See Advanced Configuration.) The range for this value is 0 to 5 (inclusive), where 0 produces the least amount of output, and 5 produces the highest.

The RoamServer software is capable of logging information about both access attempts and accounting transactions. When placed into debug mode, the RoamServer will log transactional information into a local file which can be used in troubleshooting. In addition, the software can be configured to log accounting data to either a local file or to forward it to a remote accounting server. (Previous version of the RoamServer software could log to both a local server and remote server at the same time, but this feature is not present in RS 5.0)

If your Debug value is set to any value greater than 0, you will need to customize the log file rotation and backup process so that the logs don't build up unnecessarily. A *debug_level* of *5* produces a great deal of output. This can cause the *roamserver.trace* file to grow very large, and may slow the processing time of the RoamServer. iPass recommends a debug level of 0 in a production environment.

See also Log File Deletion and Error Messages.

Page Top

Accounting Log File Configuration

The RoamServer software can be configured to write accounting information to a log file in a manner similar to that for debugging described above. The log file rotation and backup process can be customized to suit your networking environment and business needs. Depending on the type of AAA used, the RoamServer can utilize either local accounting logging or remote accounting logging. (Previous version of the RoamServer software could log to both a local server and remote server at the same time, but this feature is not present in RS 5.0).

Local Accounting

For authentication protocols that do not have a built-in remote accounting server (i.e. Unix, SITE and LDAP), the RoamServer can be configured to keep detailed local accounting records at a location specified by the user. For authentication protocols which have a remote server capable of handling accounting transactions (i.e. RADIUS, TACACS+), the RoamServer can forward the accounting record to the remote server for logging.

To configure RoamServer to log to a local accounting file:

1. Run ipassconf.csh
2. At the line, *Do you wish to add a new Acct Server*?, enter *Yes*.
3. If you wish to log accounting records to a local file, for Protocol, enter AcctFile.
4. Enter the path and name of your accounting file, or press Enter to use the default path.
5. After running the script, restart your RoamServer.

Remote Accounting (RADIUS and TACACS+ users)

Customers who have a remote server capable of handling accounting transactions (i.e. RADIUS, TACACS+) can forward the records to the remote server for logging,

To configure RoamServer to forward accounting records to your remote AAA server:

1. Run ipassconf.csh
2. At the line, *Do you wish to add a new Acct Server*?, enter *Yes*.
3. For Protocol, enter RADIUS or TACACS as appropriate.
4. Enter the details of the AAA server, as requested.
5. After running the script, restart your RoamServer.

If a remote accounting server is unreachable for some reason, accounting data that was supposed to be forwarded to it can be stored in a local file until the remote server is reachable again. The data is stored in binary format in a file called <iPass_Home>/logs/failedAcct.

To resend the data, run the script called resendacct.csh. This forwards the failedAcct file to the AAA server and then deletes the file.

Page Top

VI: Advanced Configuration

You can set other properties to take advantage of the advanced functionality of RoamServer. Some of these properties are described in detail here. These cannot be set using ipassconf.csh; instead you will need to set these properties using a text editor.

Property Help

You can obtain help on any property, including those listed here, by using a tool called config_help.csh, found in your *<iPass_Home>/bin* directory.

To list all server properties. run: config_help.csh listall

To describe usage of an property, run: config_help.csh help <property name>

Properties

This section describes a selection of properties found in the *iPassRS.properties* file.

Ascend Data Filters (AscendDataFilterN) : For Non-VPN Access

Some network providers on the iPass network filter port 25 traffic (SMTP), in an effort to prevent the distribution of spam mail on their networks. Although port 25 traffic is blocked from these providers, they allow port 25 traffic to pass to a limited number of IP addresses to allowing users to send SMTP mail to valid mail servers. The IP addresses to which port 25 traffic is allowed is communicated by the Ascend Data Filter attributes, which are sent when the user successfully authenticates. These attributes are configured in the RoamServer properties file. (The format is quite similar to how a RADIUS server's users file would be configured to return those attributes.)

Note: if users will be connecting through a VPN, this attribute can be ignored with no effects. If users will *not* be connecting through a VPN, then iPass strongly recommends you configure these settings to reflect your SMTP server(s).

Sample Settings

```
AscendDataFilter1 = ip in forward tcp est
AscendDataFilter2 = ip in forward dstip xxx.xxx.xxx.xxx/yy
AscendDataFilter3 = ip in drop tcp dstport = 25
AscendDataFilter4 = ip in forward
```

The "xxx.xxx.xxx.xxx/yy" would be replaced by an IP mask identifying the customers mail server IP address(es). For example, "218.239.99.139/32". Note that most providers only allow masks ranging from 24 to 32.

For example, if your SMTP servers' public IP address is 236.14.5.70, then the settings would look like this:

```
AscendDataFilter1 = ip in forward tcp est
AscendDataFilter2 = ip in forward dstip 236.14.5.70/32
AscendDataFilter3 = ip in drop tcp dstport = 25
AscendDataFilter4 = ip in forward
```

Note that a either a single IP address (236.14.5.70/32) or a range of IP addresses (236.14.5.0/24) can be specified.

In this other example, there are two entries. The first is a single SMTP server, and the second is a network range. Port 25 traffic will be allowed to the single IP address specified in filter2, as well as the entire network specifed in filter 3.

```
AscendDataFilter1 = ip in forward tcp est
AscendDataFilter2 = ip in forward dstip 236.14.5.70/32
AscendDataFilter3 = ip in forward dstip 236.16.6.0/24
AscendDataFilter4 = ip in drop tcp dstport = 25
AscendDataFilter5 = ip in forward
```

Up to 17 different IP addreses or range strings can be specified in this manner.

Log File Deletion(LogDirFileDeletionAge)

Log files and accounting files can grow to unmanageable sizes.To control this, you can set log files to be deleted after a specified period of time.

Default Value: 90 <days>

Routing by Realm (RouteByRealm)

Routing by realm allows routing requests to specific AAA servers, based on the user's realm or domain. Routing can also be done by routing prefix.

This allows you to use different types of authentication server, if necessary. For example, you could use both a RADIUS server and an LDAP server simultaneously. Requests from one domain, or with one prefix, can be directed to one server while requests from another domain or with another prefix can be directed to a second server.

Property : RouteByRealm

Default Value: YES

Valid Range: boolean

If routing by realm is enabled on your RoamServer, you will also need to set other properties to specify your other AAA servers, including

- Routing RealmN
- Realm
- Realm Type (1=domain, 2=routing prefix)
- AuthServerN
- AcctServerN

Sample Settings

```
RouteByRealm=YES

RoutingRealm1=Realm=mydomain.com,RealmType=1,AuthServer1=AuthServer1,AcctServer

RoutingRealm2=Realm=XY,RealmType=2,AuthServer1=AuthServer2,AcctServer1=AcctServ

RoutingRealm3=Realm=DEFAULT,AuthServer1=AuthServer1,AcctServer1=AcctServer1
```

Duplicate Filtering (FilterRequest)

This property determines the time that a user is kept in a local authentication cache, to prevent duplicate authentication requests. When using Windows NT authentication, duplicate authentication requests could cause the user to be locked out of their account. When authentication fails, a duplicate AAA request is not sent to the AAA server, but instead authentication is performed on cached users.

Property : FilterRequest

Default Value: 0

Valid Range: 0 to 10 minutes.

To disable: Set to 0.

Remote Control Messages (ControlMessageIpN and AcceptRemoteControlMsgs)

These properties allow you to configure which IP addresses your server will allow control messages from. These messages are sent over SSL and can include commands sent using rs_command, such as shutdown. By default, no IPs are enabled except the local host. If you wish to allow other RoamServers, or iPass Customer Care to send remote control messages, you will need to configure them in *iPassRS.properties.*

AcceptRemoteControlMsgs

When set to Y, this property allows remote control messages from other servers.

Default Value: NO

Valid Range: Boolean

To enable: Set to YES.

ControlMessageIpN

Each instance of this property (ControlMessageIp1, ControlMessageIp2, etc.) contains the IP of a server from which the RoamServer will accept remote control messages.

Default Value: Local host.

Valid Range: any valid IP.

To disable: set the AcceptRemoteControlMsgs (above) to N.

Automatic Software Updates

AutoUpdate

If AutoUpdate is enabled, RoamServer will check for any updates to the RoamServer software, download and install them automatically, then restart.

Default Value: NO

Valid Range: Boolean

To enable: set to YES.

UpdateInterval

This is the weekly time of day at which RoamServer will check for any updates.

Default Value: Monday 02:00

Valid Range: <any day> <24 hour time>

To enable: set AutoUpdate to YES.

Page Top

VII: Testing

There are three tests that should be performed following every installation and configuration of the RoamServer to ensure proper functionality:

1. Running the checkipass.csh tool
2. Running the RoamServer Test Tool
3. Testing with iPassConnect

When testing your RoamServer, it is recommended that you perform all of these tests in the order that they are presented here. Depending on the complexity of your system, it may take less or more troubleshooting to confirm that all is working properly.

Test 1: checkipass.csh

The checkipass test is a simulated request from the RoamServer to the AAA server, which stays local to your network. To test the RoamServer using the checkipass test, you will need to run the checkipass test program as user root. This test simply verifies that the RoamServer can authenticate a local user by communicating with the AAA server. This procedure only tests the RoamServer. No realm should be attached to the user name unless it is required by your AAA. The authentication request goes from the checkipass test to the RoamServer, then to the AAA server for authentication, and finally back to the RoamServer and checkipass program.

This test program is found in the test subdirectory of your iPass directory. You will need to use a valid user name and password for the machine on which the RoamServer is installed.

Usage: ./CheckIpass [options] -u <userid>

-p <password>	Specifies the user password.Otherwise, you will be prompted for it.
-host <host_name or ip>	Host Name or IP Address to send request to.
-port <port number>	Port number of the destination host.
-type <request type>	Request type. Default is normal. Choices are: auth, acct, start, stop, all, normal.
-timeout <timeout>	Timeout, in milliseconds, to wait for a reply. Default is 30000 milliseconds.
-aaa_auth_server <number>	Specify which AAA Authentication Server should handle the request from the RoamServer.
-aaa_acct_server <number>	Specify which AAA Accounting Server should handle the request from the RoamServer.
-show_radius_attrs	Shows the list of supported RADIUS attributes.

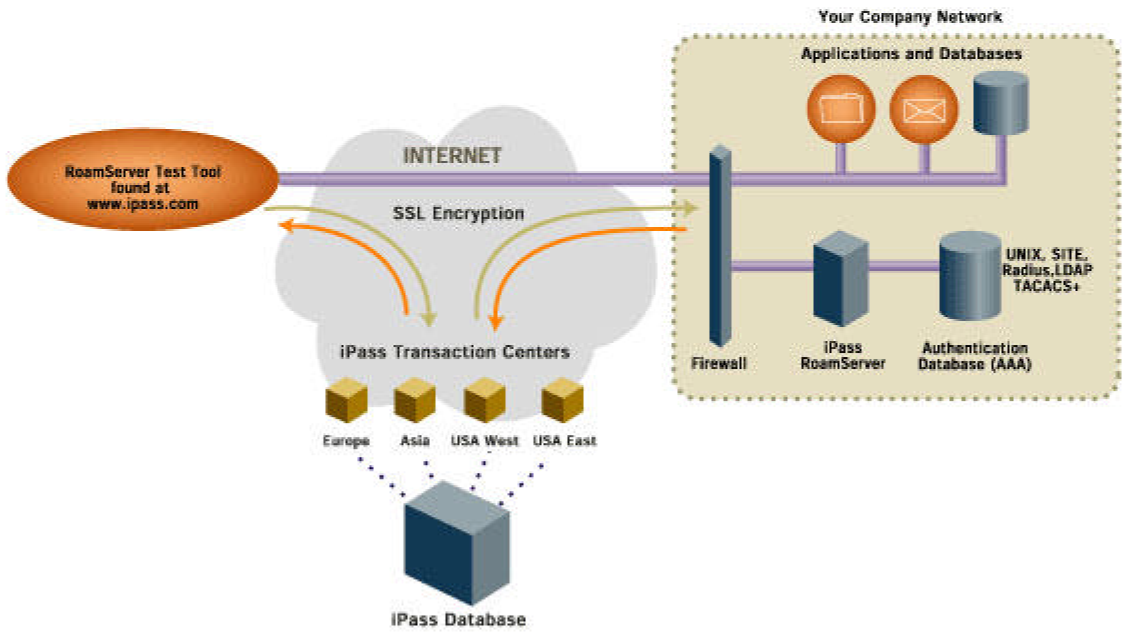
-interactive :	Run this program in interactive mode.
-help	Show the help/usage of this program.

You will receive output stating the status of the checkipass test = Accept or Reject. If *status = Accept* then the RoamServer is properly installed, configured and working, and you may proceed to the next test.

Due to the simplicity of this test, a Reject test result should isolate the problem to your local server and reduce troubleshooting efforts. Possible causes for a failure here include:

- *Invalid user name or password.* The user in this test must have local login privileges to that system.
- *Invalid certificate.* If the certificate is corrupt, then it will need to be replaced. You can verify the dates and readability of your certificate by running the tools *view_certificate_dates* and verify_certificate in your <iPass_Home>/bin directory . Generally, if the certificate is readable, then it is not corrupt.
- *Improper configuration.* Verify that you have correctly entered all of the information in the setup program and that your server is running on port 577.
- *For RADIUS users, invalid shared secret.* Verify that your shared secret is entered properly.

Test 2: RoamServer Test Tool


Your Company Network
Applications and Databases
RoamServer Test Tool found at www.ipass.com
INTERNET
SSL Encryption
iPass Transaction Centers
Europe
Asia
USA West
USA East
iPass Database
Firewall
iPass RoamServer
Authentication Database (AAA)
UNIX, SITE, Radius,LDAP TACACS+

The RoamServer Test Tool extends the verification performed in the checkipass test by sending a simulated authentication request across the iPass network. In this test:

- An authorization request is generated by the tool and sent directly to an iPass Transaction Server, where the user name, domain and password are verified as belonging to a valid account in the iPass database.
- The Transaction Server forwards this authentication request to the Primary RoamServer at your company on port 577.

- The RoamServer receives the request, drops the domain name, and either authenticates locally (Unix or SITE), or forwards the request to your AAA server (RADIUS, LDAP or TACACS+)
- Upon successful authentication, the request is relayed using SSL encryption back to the RoamServer Test Tool.

Unlike the checkipass test, in which the local network user name and password were used, in this test you will need to provide a user name and domain name that are specific to the iPass network.

This test is available as a Web-based tool, and can be reached from the iPass Web site.

To run the tool:

1. Point your browser to www.ipass.com.
2. On the sidebar, click *Login*.
3. Enter your iPass Company ID (the company code used for your installation), user name and password and click the *Login* button .
4. On the sidebar, click *Customer Care.*
5. Click *Network Status Pages*.
6. Click *RoamServer Test Tool.*
7. Enter your iPass user name (with domain name) and password in the fields provided and click the *Submit* button.
8. This test will display a lot of output. Scroll down to the bottom to look for an *Accept* or *Reject* response before viewing the rest of the results.

Successful results in this test ensures that any user authorized to access your system can now roam on the iPass Network.

In addition to performing this test with a legitimate user name and password, you should also run the test with an invalid user name and/or password to ensure that the authorization attempt will be rejected.

Test 3: Dial-in Test Using iPassConnect

The final test to perform is an actual dial-in test using the customized iPassConnect application that was sent to you. If you do not yet have your customized iPassConnect , inform your iPass technician, or fill out the iPassConnect customization form on the iPass Web page, and one will be built for you.

If all tests are successful, this completes the RoamServer installation. Congratulations and thank you for beginning your relationship with iPass, Inc!

Page Top

VIII. Running RoamServer

Starting RoamServer

To start your RoamServer: in the <iPass_Home>/bin directory, run:

```
roamserver_start.csh &
```

Some systems will shut down all processes started by a user when that user logs off. If this is the case, run

```
nohup roamserver_start.csh &
```

Shutting Down RoamServer

To shut down RoamServer: in the <iPass_Home>/bin directory, run

```
roamserver_stop.csh.
```

RS_Command

You can also perform many runtime functions by using the tool RS_command, in the <iPass_Home>/bin directory.

Usage: `rs_command <command options>`

The default port for rs_command is the standard RoamServer default port, 577.

Command Options

`-host <IP address>`	Specifies the IP address of the machine to send the command to.
`-port <port number>`	Specifies the server port number to send the command to. Default is the local server's listener port (577).
`-shutdown`	Causes the server to shutdown.
`-restart`	Causes the server to restart.
`-software_update`	Causes the server to do a software update.
`-reload_config`	Causes the server to reload many (but not all) of the properties from the ipassRS.properties file. These are: • AutoUpdate flag, used to enable/disable automatic software update. • AAA Servers (AuthServer and AcctServer properties) • Policy Rules, if feature is enabled. • RS Monitor configs for the HeartBeat Messages.

	• Log Rotation parameters. • DebugLevel of server. For a complete reload, you should use the -restart switch.
-dump_queue	Causes the server to dump the queue elements to a file.
-get <filename> -host <IP address> -port <port number>	Get a file from a remote RoamServer. • Use filename RS.properties to get the RoamServer properties file. • Use filename RS.trace to get the RoamServer trace file. • Optionally, use any valid filename relative to the RoamServer home directory.
-post <Name=value;Name1=value1> -host <IP address> -port <port number>	To post configuration changes on a remote host., where Name=Value pairs are the properties settings separated by ";", <IP address> is the IP address of the remote host, <port number> is the port number of the remote host.
-post_file <file> -host <IP address> -port <port number>	To post configuration changes on a remote host, where <file> contains the configuration changes to be uploade d to the RoamServer, <IP address> is the IP address of the remote host, <port number> is the port number of the remote host.
-version	Print the server version.

Java Options

-Dipass.server.type=RoamServer	Specifies the iPass Server Type to send the command to.
-Dipass.server.home=install-directory	Specifies the iPass Installation Directory.
-Dipass.config.file=config-file-name	Specifies the properties file to load from.
-Dipass.ipass.trace.file=log-file	Specifies the file to log trace (debug) messages to.

Page Top

Appendix 1: The ipassLDAP.properties Configuration File

How iPass LDAP authentication works:

Action 1: Exact Match will be performed to authenticate the user. That means a bind to the LDAP server using an exact match DN and the users password. The exact match DN is comprised of the login username attached with any base DN specified in the *ipassLDAP.properties file*. The user portion (Relative DN) of the exact match DN is by default uid=username, but can be customized with the *LdapExactMatchRdn* configuration in the *ipassLDAP.properties file*.

The exact match operation can be disabled by setting 'LdapDoExactMatch = no' in the *ipassLDAP.properties file*.

Action 2: Anonymous bind and search will be performed to authenticate the user. That means a bind to the LDAP server using a NULL userid and password. If anonymous binds are not supported by the LDAP server, a *LdapBindDn* and *LdapBindPasswd* can be specified in the *ipassLDAP.properties file*.

After a successful bind, we search the LDAP directory for the user starting from a base DN as specified by the *LdapBaseDn* and filtering with the *LdapSearchFilter*. If a user (and only 1 user) is found during the search, a simple bind to the LDAP server will be performed to authenticate the user. This last authentication will be done using the DN of the user found during the search and the password supplied at login time.

The anonymous bind and search will NOT be performed if the user was authenticated during the exact match, or if no *LdapBaseDn* was specified in the *ipassLDAP.properties file*.

Overview of user configurable options in ipassconf file:

Option	Default value	Example values
ipassLDAP.properties file	<no file>	<iPass_Home>/ipassLDAP.properties

Note: When options are not specified in ipassRS.properties file, the default values will be used.

Option: *ipassLDAP.properties file*

Description: Configuration to specify a file containing special LDAP customizations, such as *LdapBaseDn, LdapSearchFilter, LdapDoExactMatch, LdapExactMatchRdn, LdapBindDn* and/or *LdapBindPasswd* (which are described in the 'Overview of user configurable options in *ipassLDAP.properties file* file' section below). All lines in the *ipassLDAP.properties file* that begin with a space or # sign will be ignored.

Default value: <no file>

Example: ipassLDAP.properties file = <iPass_Home>/ipassLDAP.properties

Overview of user configurable options in "ipassLDAP.properties file" file:

Configuration	Default value	Example values
LdapBaseDn	NULL	o=company.com
LdapSearchFilter	uid=$USERID	mail=$USERID@$DOMAIN
LdapDoExactMatch	NO	YES
LdapExactMatchRdn	uid=$USERID	mail=$USERID@$DOMAIN
LdapBindDn	NULL	binddn
LdapBindPasswd	NULL	bindpasswd
LdapCompareAttr	NULL	SomeUserAttribute
LdapSearchMoreServers	no	yes

Note: When an *ipassLDAP.properties file* is not present or if an option in not specified in that file, the default values will be used.

LDAP Configuration: LdapBaseDn

Description: Configuration to specify base DNs to be used during LDAP authentication. When configured, it will be appended to the *LdapExactMatchRdn* during exact match bind and used as a search base during the LDAP search operation.

Important: If no **LdapBaseDn** is configured, then no anonymous bind and search will be performed.

Default Value: NULL

Examples:
LdapBaseDn = ou=people,o=company.com
LdapBaseDn = o=company.com
LdapBaseDn = dc=company,dc=com

Note: Multiple base DNs (more than one line) are permitted in the *ipassLDAP.properties file*. When multiple base DNs are configured, the authentication process will use them in the order they appear in the *ipassLDAP.properties file*. If authentication fails using the first **LdapBaseDn**, authentication will be re-attempted using the second **LdapBaseDn** and so on.

LDAP Configuration: LdapSearchFilter

Description: Configuration to set a custom filter when searching an LDAP server for a user. If this option is not set, the default filter (uid=$USERID) will be used. When an exact match bind is disabled or is unsuccessful, an anonymous bind and search will be used. A custom filter may be supplied for the search. Any variables supplied in the format of $VARIABLE will be replaced with the actual value of that variable. The current variables supported are $USERID and $DOMAIN.

The variables' values are taken from the user's login. For example if someone logs in as joe@company.com, the variable $USERID would be replaced by "joe" (that is, everything to the left of the left most @ sign, not including any prefix such as iPass/ if the login was iPass/joe@company.com). The variable $DOMAIN would be replaced by "company.com" (that is, everything to the right of the left most @ sign).

For example: if the search filter is "(&(mail=$USERID@$DOMAIN)(dialup=true))", when "joe" from "company.com" logs on, the search filter will be converted to "(&(mail=joe@company.com)(dialup=true))".

Default Value: **LdapSearchFilter** = uid=$USERID

Examples:

LdapSearchFilter = uid=$USERID
LdapSearchFilter= mail=$USERID @$DOMAIN
LdapSearchFilter = (&(uid=$USERID)(dialup=true))

Note: Only one filter (one line) is presently allowed in the *ipassLDAP.properties file*.

LDAP Configuration: LdapDoExactMatch

Description: Configuration to disable/enable binding directly to the LDAP server for user authentication using only the user's login id, password, and any base DN by the *LdapBaseDn* configuration. Accepted values are "yes" and "no".

Default Value: yes

Example: **LdapDoExactMatch**= no

LDAP Configuration: LdapExactMatchRdn

Description: The DN used for the exact match bind is comprised of two parts: the relative DN (RDN) and the base DN. The base portion can be specified by the *LdapBaseDn* configuration. The relative DN's format can be specified by the *LdapExactMatchRdn*. The RDN is by default uid=$USERID, where the variable $USERID is replace by the username specified at login time. The current variables supported are $USERID and $DOMAIN.

For example:

User "joe" exists in a LDAP tree with a DN of "uid=joe,ou=people,o=company.com", and he logs in as joe@company.com. For a successful exact match bind, leave the *LdapExactMatchRdn* as default and set the *LdapBaseDn* to "LdapBaseDn = ou=people,o=company.com".

User "Mary" exists in a LDAP tree with a DN of "cn=Mary,dc=company,dc=com", and she logs in as Mary@company.com. For a successful exact match bind, set the *LdapExactMatchRdn* to "cn=$USERID" and set the *LdapBaseDn* to "dc=company,dc=com".

Reminder: The exact match bind can be disabled by setting *LdapDoExactMatch* to "no".

Default Value: uid=$USERID

Examples:

- LdapExactMatchRdn = cn=$USERID
- LdapExactMatchRdn = $USERID

Note: Only one *LdapExactMatchRdn* (one line) is presently allowed in the *ipassLDAP.properties file*.

LDAP Configuration: LdapBindDn

Description: For LDAP servers that do not support anonymous binds, this configuration will set a specific DN to be used for binding to the LDAP server before performing a search operation. When anonymous binds are supported, omit this configuration and the default value of NULL will be used.

Default Value: NULL

Example: LdapBindDn = uid=bindmaster,ou=people,o=company.com

LDAP Configuration: LdapBindPasswd

Description: For LDAP servers that do not support anonymous binds, this configuration will set a password to be used for binding to the LDAP server before performing a search operation. When anonymous binds are supported, omit this configuration and the default value of NULL will be used.

Default Value: NULL

Example: LdapBindPasswd = bindpasswd

LDAP Configuration: LdapCompareAttr

Description: Configuration to enable comparison of user passwords against a specific user attribute in the LDAP directory as a means of authentication. The user attribute specified must contain a password saved in clear test in the LDAP directory for *LdapCompareAttr* to work.

This compare replaces the final user bind to authenticate the user. The user bind authenticates against the standard password attribute (usually *userpassword*), which may or may not be encrypted in the LDAP directory.

Default Value: NULL

Example: LdapCompareAttr = roamingPassword

Comments about using LDAP authentication:

If a user logs on using a full DN (uid=Joe,ou=people,o=company.com), then a base DN (LdapBaseDn configuration in the *ipassLDAP.properties file*) should not be used since performance will be reduced. This is because a base DN is added on to the login name when an exact match bind is performed.

Suggested configurations in the ipassLDAP.properties file:

Example 1 (Most common):

For companies with more a elaborate LDAP directory structure where roaming users are stored in different directories:

```
uid=user1,ou=development,o=company.com
uid=user2,ou=finance,o=company.com
uid=user3,ou=marketing,o=company.com
etc.…
```

Performing a search for the user might be a simpler approach. Therefore the exact match bind step can be skipped all together. If all users login with the format of user1@company.com, then only do an anonymous bind and search of the LDAP directory.

Set the following in the *ipassLDAP.properties file*:

```
LdapBaseDn= o=company.com
LdapDoExactMatch = no
LdapSearchFilter = uid=$USERID
```

Example 2:

For companies with simpler a LDAP directory structure where all roaming users are stored in the same directory:

```
uid=user1,ou=people,o=company.com
uid=user2,ou=people,o=company.com
```

uid=user3,ou=people,o=company.com
etc.…

All users are in the "ou=people,o=company.com" directory. If all users log in with the format of user1@company.com, then to bind to the LDAP server on the first try with the exact match bind.

Set the following in the *ipassLDAP.properties file*:

```
LdapBaseDn= ou=people,o=company.com
```

Example 3:

For companies whose roaming users login with a full Distinguished Name (DN) such as: "uid=user1,ou=development,o=company.com@company.com", the user id portion (which is everything to the left of the left most @ sign) is the full DN of the user.

Only the exact match bind is needed.

Set the following in the *ipassLDAP.properties file*:

```
LdapExactMatchRdn = $USERID

LdapDoExactMatch=YES
```

Page Top

The *checkldap* Test Tool

You can check your LDAP configuration using the checkldap test tool.

Usage: *checkldap -host <hostname or ip> [OPTIONS]*

Options

-server_attr <namingcontexts, supportedldapversion, all>	Retrieve specific or all server attributes.
-port <port>	LDAP server port. Default is 389.
-bind_dn <bind DN>	The DN for binding/connecting to the LDAP server.
-bind_pw <bind password>	The Password for binding/connecting to the LDAP server.L it empty and you will be prompted for it when the bind DN i present.
-search_scope <search scope>	The search scope. Valid values are: 0 (Object Scope), 1 (OneLevel Scope) 2 (Subtree Scope). Default is 2 (Subtree Scope).
-base_dn <base DN>	The base DN specifies where on the LDAP server to begin search from.
-search_filter <search filter>	The search filter indicates what to seach for. Example: uid=

-user_pw <user password>	The user password to authenticate.Leave it empty and you be prompted for it before a search operation.
-help	Prints the help/usage of this program.

Using Active Directory

When using Active Directory, configure the RoamServer to point to any domain controller server when setting up your authentication server. AD listens on TCP port 389, but for large AD 'forests', consider configuring RoamServer to point to Global Catalog DCs on TCP port 3268. Normal LDAP traffic on port 389 to AD DCs will not support 'referral chasing' for object binds outside of the resident domain which the DC resides in. To be able to authenticate users in other domains in your organization, RoamServer needs to authenticate against a GC DC in any domain, preferably at the root of the forest.

Here is an example of an iPassLDAP.properties file configured for use with Active Directory. Red lines are mandatory fields.

```
# File: ipassLDAP.properties

#

# Contains configurations for customizing LDAP authentication

#

# Blank lines and lines beginning with # are ignored.

# Uncomment and customize the LdapBaseBn line to set        a search base.

# A minimum of 1 LdapBaseDn is required for a search        to occur.

LdapBaseDN=DC=corp,DC=ipass,DC=com

# Uncomment and set the LdapSearchFilter line to set        a search filte

# Supported variables are USERID and DOMAIN.

LdapSearchFilter=sAMAAccountName=$USERID

# Uncomment the following to disable the exact match        bind. This is

# recommended when only the LDAP search is needed.        Options: NO or Y

# LdapExactMatch = NO

# Uncomment and customize the 'exact_match_rdn' line        to specify the

# format for exact match bind. Supported variables        are $USERID and

#
```

```
    # LdapExactMatchRdn =uid=$USERID

    # Uncomment and customize the 'bind_dn' and 'bind_passwd'        lines

    # if your LDAP server does not support anonymous binds.

    LdapBindDn = CN=restricted_user_id,CN=Users,DC=corp,DC=ipass,DC=com
    LdapBindPasswd=secret

    # Uncomment and customize the 'compare_attr' line        to specify a user

    # attribute to compare the password with when authenticating.        NOTE:

    # This will replace the final user bind for authenticating.

    #

    # LdapCompareAttr= someUserAttribute
```

Page Top

Appendix II: Error Messages

This section lists error messages that can be returned by the RoamServer at Debug Levels 0, 1 and 2. Although other debug levels are possible, they are used only for packet dumps and no error messages are associated with them.

Variables denoted in the list by + (for example, *+ioe.getMessage()*) will be replaced at runtime with specific data.

Feature	**Debug Level**	**Message**
Tacacs+		
	1	Error occurred while trying to communicate to the TACACS+ server
	1	Failed to convert TACACS+ packet to bytes
	1	"Failed to open TCP socket to TACACS+ server: IO Error, "+ioe.getMessage()
	1	"Failed to open TCP socket to TACACS+ server: "+e.getMessage()
	1	Failed to send packet to TACACS+ server" +ioe.getMessage()
	1	Unexpected NULL clientSocket, socket could be closed.
	1	Timed Out reading packet from TACACS+ server " +iioe.getMessage()
	1	"Failed to read packet from TACACS+ server " +ioe.getMessage()
	1	Cannot parse raw TACACS+ packet
	1	"Error closing socket to TACACS+ " +ioe.getMessage()
	1	"ERROR parsing header of packet received from TACACS+ server"
	1	"Unsupported reply packet type " +this.hdr_type +" received from TACACS+ server"
	1	"ERROR decrypting TACACS+ packet"
	1	"ERROR: missing TACACS+ packet type"
	1	"parse() not supported for this reply packet type " +pktType

	1	"ERROR: missing TACACS+ packet type"
	1	"ERROR: toBytes() not supported for packet type " +pktType
	1	"ERROR encrypting TACACS+ packet"
	1	"CHAP challenge conversion failed."
	1	"CHAP password conversion failed."
	1	"ERROR encrypting TACACS+ packet"
	2	Error or Timeout in getting reply from TACACS+ server
	2	Password is NULL, TACACS+ Minor Version 0 does not support CHAP authenticatio
	2	Error/Timeout getting first auth reply from TACACS+ server
LDAP		
	0	"Server's LDAP Info is Missing "
	0	"Unexpected return code (" +rc +")"
	0	"Internal Error: LDAP server address not set"
	1	"Illegal LDAP Configuration: Must configure an "+LdapInfo.LDAP_BASE_DN +" or E "+LdapInfo.LDAP_DO_EXACT_MATCH
	1	"Error creating RDN from ldapExactMatchRdn"
	1	"ExactMatchBind failed " +ne.getMessage()
	1	"Error creating Search Filter."
	1	"LDAP Authentication failed " +reason
	1	"Error, LDAP search found multiple matchs "+entryCount +" found for this user"
	1	"LDAP Search found multiple matches for this user " +slee.getMessage()
	1	"LDAP Search exceeded " +searchTimeout +" millisecond time limit: " +tlee.getMess
	1	"LDAP Search Error: " +ne.getMessage()
	1	"LDAP Compare of (" +name +") attribute with password failed."
	1	"LDAP Compare of (" +name +") attribute failed: " +ne.getMessage()
	1	"Unexpected NULL ldap context"
	1	"Invalid attribute name: "+attrName+", in line: "+origString
	1	"Could not authenticate user at this LDAP server"
	1	"TIMEOUT while talking to LDAP server after " +sInfo.NumRetry +" tries"
	2	"Error while closing connection to LDAP server" +ne.getMessage()
SSLPost		
	0	fileDesc+fileName+" does not exist"
	0	"Cannot read "+fileDesc+filename
	0	"Failed to instanciate SSLPostCommunicator: "+cce.getMessage()
	0	"Could not instantiate SSLSocketImpl"
	0	"ERROR: Missing IpassDictionaryEntry"
	1	"Socket receive timed out"
	1	"Failed to receive data from server: " + serverInfoRec.IpAddress + ":" + serverInfoRe
	1	"IOEXCEPTION: while talking to server: " + serverInfoRec.IpAddress + ":" + serverInfoRec.Port
	1	"received null Communicator object"
	1	"received null serverInfoRec"
	1	"received null requestPkt"
	1	"received null replyPkt"
	1	"Could not create sslSocket: doHandshake failed"
	1	"Could not create sslSocket: Instantiation failed"
	1	"sslSocket null for ServerSide communicator"

	1	"Could parse post packet: " +replyStr
	1	Malformed Post Packet
	1	"Malformed post packet header"
	1	"Unexpected NULL sslSocket."
	1	"Error parsing MultiInstance attribute "+name+", of type " +de.getType()
	1	"Error parsing attribute "+name+", of type " +de.getType()
	1	"Error in converting the packet to bytes: " + e.toString()
	1	"Error for attribute "+name+ ": "+i.getMessage()+" Ignoring it"
	1	"Dropping attribute for ipassCode " +ipassCode+" value "+value+", NumberFormatException: "+nfe.getMessage()
	1	Base64 Decode ERROR: Dropping OBJECT of ipassCode " +ipassCode+" value "+v
	1	"Dropping OBJECT of ipassCode " +ipassCode+" value "+value+", OptionalDataExc "+o.getMessage()
	1	"Dropping OBJECT of ipassCode " +ipassCode+" value "+value+", ClassNotFoundException: "+c.getMessage()
	1	"Dropping OBJECT of ipassCode " +ipassCode+" value "+value+", IOException: "+i.getMessage()
	1	"Dropping attribute for ipassCode " +ipassCode+" value "+value+", NumberFormatException: "+nfe.getMessage()
	2	"NULL sslServerSocket, listener socket could be closed."
	2	"SSL handshake failed, closing accepted socket."
	2	"Listeners are shutdown, closing accepted socket."
	2	"Rejecting packet from: " +sslSocket.getHost()
	2	"Error: No ipassPkt to send"
	2	"Unexpected NULL sslSocket, socket could be closed."
	2	"Could parse post packet: " +packetStr
	2	"Error parsing IpassPostPkt: Unknown URI/request type " +uri
	2	"Error parsing IpassPostPkt: missing empty string."
	2	"Error parsing IpassPostPkt."
	2	"Unknown PostPkt attribute (" +name +"): ignoring it."
Handlers		
	0	"Software update failed"
	0	"Download failed"
	0	"Error occurred while trying to instantiate RSPolicyRules: " + i.getMessage()
	0	"Error occurred while adding policy rule: An entry with the same rule:" + id + " exists!
	0	"File "+policyFile+" not found"
	0	"Failed to Shutdown due to policy errors as the TransactionController is null"
	0	"Failed to Shutdown due to policy errors as the TransactionContext is null"
	0	Can not find TRANSACTION CONTROLLER
	0	Can not find exceptionHandler
	0	Could not get LOCAL_HOST_IP
	0	Error occurred while trying to instantiate " + s.toString()
	0	Error occurred while trying to send the reply packet
	0	No Server found for the following transaction type: "+ reqTypeName
	0	No valid handler found for the request of type "+type);
	0	ERROR occurred while trying to save the acct record in a file: "+i.getMessage())
	0	Error occurred while trying to instantiate RSAcctReqHandler: " + s.getMessage()

	0	Unexpected ERROR: "+Config.FAILED_ACCT_LOG_DIR+" property not set!
	0	Could not create directory \"" + failedDirPath + "\" to store failed accounting records.
	0	ERROR, expected "+Config.FAILED_ACCT_LOG_DIR+" to be a directory, got \"" + failedDirPath + "\" instead.
	1	"Software Update Failed due to failure to load the Server's Version Table."
	1	"Unable to copy "+this.serverJarFileName +" to "+this.updatefilesJarFileName
	1	"User " + user_id + " is denied access based on the policy rule: \"" + id + "\" "
	1	"IO error in loading policy File "+policyFile
	1	"Error loading the policy file"
	1	"Cannot get SSLPOST listener port, defaullting to:" + UNKNOWN_PORT
	1	"Failed to handle Heartbeat message!"
	1	"Failed to load RS Policy Rules: "+se.getMessage()
	1	"Policy Restriction. Verify Policy Failed."
	1	"Authentication Rejected: Invalid Reply Packet"
	1	"ERROR: list lock is NULL. Cannot check for duplicates in our accessList"
	1	"exception ocurred: " + e.toString()
	1	"ERROR: list lock is NULL. Cannot add entry to our accessList"
	1	"No such hashing alrorithm error: "+nsae.getMessage()
	1	handleRequest-Communicator object is null
	1	Error: File: " + fileName + " does not exist on the server
	1	Error: File: " + fileName + " content is empty!
	1	failed to get file contents
	1	Invalid Request: Failed to get the path of the file: " + fileName
	1	Invalid Request: Cannot return the files in the keys folder!
	1	Invalid Request: filename is not from the $ipass.server.home: " + fileName;
	1	Invalid Request: File:" + fileName + " does not exist on the server!
	1	"Invalid Request: File name not specified!
	1	handleRequest-Communicator object is null
	1	Failed to reload the new config file, reverted to the old config file...
	1	Invalid request, Failed to Reload the new config file, and failed to rename " + fileNar ".bak to " + fileName + "\nPlease copy the " + fileName + ".bak to " + fileName + " ar restart the server!
	1	Invalid request, Failed to Reload the new config file, and failed to find the " + fileNar ".bak in order to rename it to " + fileName + "\nPlease copy the " + fileName + ".bak fileName + " and restart the server!
	1	Invalid request, Failed to Reload the new config file, and failed to delete it.\nPlease c " + fileName + ".bak to " + fileName + " and restart the server!
	1	Failed to rename " + fileName + " to " + fileName + ".bak"
	1	Failed to delete " + fileName + ".bak"
	1	Error, Config Filename could not be obtained!
	1	source Ip is null, not a valid CTRL_MSG_IP
	1	netSourceIp +" is not a valid/configured CTRL_MSG_IP
	1	Invalid Request: File contents are empty!
	1	Invalid Request: Failed to load the config changes: " + e.getMessage()
	1	Protocol is not supported by current version of software: Server ID=" + serverInfoRec.ServerInfoId + ", Server Protocol= " + serverInfoRec.AuthProtocol);
	1	ERROR: Cannot get communicator for server IP: " + serverInfoRec.IpAddress + ", o Protocol: " + serverInfoRec.AuthProtocol
	1	"No Servers found: Null returned from getRoute()"

	2	netSourceIp +" is not a valid/configured CTRL_MSG_IP");
RADIUS		
	0	Failed to open DatagramSocket
	0	Cannot get LOCAL_HOST_IP, unable to set NAS_IP in RADIUS packet
	0	IOException on listener for port "+serverPort+": "+e.getMessage();
	0	IOException on listener for port is due to RADIUS Listeners being shutdown
	0	ERROR creating the UDP socket at port "+port+". (Port may be in use)");
	0	Failed to instanciate SharedSSLPostCommunicator
	1	Unexpected NULL socket, socket could be closed
	1	IOException on DatagramSocket
	1	Error occurred while trying to talk to AAA server
	1	Failed to communicate with radius server after " +sInfo.NumRetry +" tries
	1	RADIUSPkt parsing errors
	1	Input not a byte array
	1	Empty RADIUS data
	1	Illegal type in RADIUS packet
	1	Missing identifier in the RADIUS packet
	1	Missing Length in the RADIUS packet
	1	Missing authenticator in the RADIUS packet
	1	Missing code in the RADIUS packet
	1	Missing length in the RADIUS packet
	1	ERROR: Invalid CHAP_PASSWD lenght of "+dataLen
	1	ERROR: Invalid MESSAGE_AUTHENTICATOR lenght of "+dataLen
	1	Missing IpassDictionaryEntry for radius code " + code
	1	Illegal data type
	1	Malformed radius packet (When data length is longer than the packet header specifi
	1	ERROR: missing MESSAGE_AUTHENTICATOR to validate EAP-Message
	1	ERROR: missing Request Authenticator to validate EAP-Message
	1	ERROR: failed to re-calculate Message-Authenticator"
	1	ERROR: Invalid Message-Authenticator
	1	ERROR: missing Request Authenticator
	1	ERROR: failed to generate test Authenticator
	1	ERROR: missing Response Authenticator
	1	ERROR: Invalid Response Authenticator
	1	No such algorithm
	1	Digest Exception
	1	No valid RADIUS code for Ipass Packet Type "+getPktType()+" Status "+status
	1	Missing IDENTIFIER header attribute, using value of "+ident+" instead
	1	Error: CHAP Identifier missing from packet
	1	CHAP password conversion failed.
	1	CHAP challenge conversion failed.
	1	ERROR: missing Shared Secret to calculate the Message Authenticator
	1	ERROR: when calculating HMAC digest of Message Authenticator
	1	ERROR: Request Authenticator is missing.

	1	Unsupported encoding exception
	1	NoSuchAlgorithmException
	1	Exception: " + e.toString());
	1	ERROR: missing Shared Secret
	1	ERROR: Base64 Decode of iPass Attribute " +ipassAttrCode +" failed
	1	WARNING: Unable to get Dictionary entry for iPass Attribute
	1	ERROR: UTF8 conversion of iPass Attribute " +ipassAttrCode +" failed
	1	ERROR: Base64 Decode of iPass Attribute " +ipassAttrCode +" failed
	1	ERROR: Base64 Decode Vendor Specific Attribute " +vendorId+":"+vendorType +" failed
	1	ERROR: Invalid Vendor Specific Attribute format
	1	Vendor ID missing from Vendor Specific Attribute
	1	Vendor Type missing from Vendor Specific Attribute (VendorID="+vendorId+
	1	Vendor Length missing from Vendor Specific Attribute (VendorID="+vendorId+", VendorType="+vendorType+
	1	Value missing from Vendor Specific Attribute (VendorID="+vendorId+", VendorType="+vendorType
	1	Value from Vendor Specific Attribute is corrupted. (VendorID="+vendorId+", VendorType="+vendorType realLen="+readLen+",
	1	expected len was "+vendorValueBytes.length
	1	Cannot convert attribute "+attr +", RADIUSType type of IPADDRESS to iPass type " iPassType
	1	Cannot convert attribute "+attr +", RADIUSType of Integer to iPassType " +iPassTyp
	1	Unsupported iPass attribute " +attr +", with radius value " +radiusValue
	1	NULL input: key is null
	1	NULL input: text is null
	1	Hashing error
	1	No such hashing alrorithm error
	2	Cannot parse raw packet
	2	Receive timeout set to " +sInfo.IdleTimeout milliseconds
	2	RADIUSBufferSize error
	2	NULL serverSocket, listener socket could be closed.
	2	Started RADIUS Listener "+i +" on port "+listenerThreads[i].getServerPort());
	2	Cannot convert attribute "+attr +", RADIUSType of TEXT to iPassType " +iPassType
	2	Unsupported String Encoding: " +attr +", with radius Type " +radiusType
	2	Cannot convert attribute "+attr +", RADIUSType of String to iPassType " +iPassType
	2	Cannot convert to Integer: "+attr +", with radius Type " +radiusType
	2	Cannot convert attribute "+attr +", RADIUSType Time to iPassType " +iPassType
	2	Cannot convert attribute "+attr +", RADIUSType BYTEARRAY to iPass type " + iPas
	2	Illegal data type " + radiusType
Site		
	0	Failed to load SiteCommunicator library
	1	Error occurred while trying to do Site file authentication
	2	Failed talking to SITE server
Unix		
	0	Failed to load UnixCommunicator library
	1	Error occurred while trying to do UNIX authentication
	2	Failed talking to Unix server

NT and NT RAS		
	2	Received authentication accept packet from Windows Server
	2	Received authentication reject packet from Windows Server
AcctFile		
	1	Failed to write to local AcctFile
	1	Error occurred while trying to talk to Windows server
	1	Failed talking to Windows server
	2	Received unexpeted null packet when writing to local AcctFile

Page created by: Michelle Brahm, Technical Writer
Updated for RoamServer 5 by: Ed Messina, Technical Writer
Content contributed by:
David Gutierrez, Network Access Engineer
Updated Content for RoamServer 5 from Pierre Beehler, Peter Crowdell and Steve Cirimele