NetServer 6.0.0 Administrator Guide

iPass NetServer is designed to receive access request packets from a Network Access Server using the RADIUS protocol, and route them through the iPass network. The NetServer 6.0.0 Administrator's Guide provides instructions for installation, configuration and operation of NetServer at an iPass network provider site.

Main Topics

NetServer Printable Administrator's Guide

The NetServer Printable Administrator's Guide is not an interactive PDF. Its function is strictly for printing.

NetServer Architecture

The Authentication Cycle

Access requests sent over the iPass network travel a complete cycle from remote endpoint to corporate sites. The complete cycle, illustrated here, works as follows:

1. A remote user connects to an iPass-enabled network provider with the Open Mobile client.
2. The request is sent using the RADIUS protocol to a Network Access Server (NAS) at the provider site, where it is authenticated against the local AAA server and determined to belong to an iPass customer.
3. Depending on the configuration, either the NAS or the AAA forwards this information through RADIUS (UDP) to the NetServer, which sorts the requests and identifies valid iPass users. These packets are translated into the iPass protocol using Secure Sockets Layer (SSL) encryption before the NetServer transmits them to one of the iPass Transaction Centers.
4. The iPass Transaction Center verifies if the NetServer from which it received the request is configured as a valid source IP address in its database. If not, it rejects the request.
5. The Transaction Center records the user's authentication request and examines the realm to determine whether it is registered to an iPass customer account. If the realm is valid, the user's credentials are forwarded to an iPass RoamServer at the associated provider or corporation for authentication.
6. At the corporate or provider site, the RoamServer receives each user authentication or accounting request, decrypts, and translates the packet to the native authentication protocol (RADIUS, TACACS+, LDAP, etc), and forwards it to the local AAA server for authentication and authorization.
7. After the AAA server has authenticated the user, the response packet is sent back to the RoamServer to be re-encrypted, before it is returned through SSL to the iPass Transaction Center and back to the NAS at the iPass provider site where the request was initiated.
8. If the session is authorized, the provider's NAS establishes a PPP session, assigning the user an IP address, default gateway, and a DNS server address, granting access to the Internet.
9. To access resources behind the company's firewall, the remote user initiates a virtual private network (VPN) client and enters a second password to obtain authorization for access to the corporate network. Once authorized, the VPN creates a tunnel between the user and the corporate network to allow encrypted data to travel securely over the Internet.

Go to: **Other Product Documents** > **NetServer Admin Guide**

Installation

This section contains instructions on how to install or upgrade NetServer. The following details are available:

Preparation

Before installing NetServer, you should have already installed your RADIUS AAA server or NAS, and configured and tested the appropriate databases to authenticate your own local users.

You should have the following information:

- The IP address of the host on which you plan to install NetServer. This should be the Public IP Address registered to your iPass account which will be used to connect to the iPass AAA fabric. Your certificate cannot be validated until iPass places this IP address into our database.
- The IP address, port numbers and secrets of RADIUS servers which forward requests to the netserver.
- The iPass ISP Code given to your company when it signed up with iPass. If you do not have this code, please contact your iPass NetServer Installation Engineer.
- Your username and password for the iPass FTP site where you will download the software. If you do not have these credentials, please contact your iPass NetServer Installation Engineer.

In addition, you should make sure that you have access to the following:

- A Mail Transfer Agent (such as Sendmail) installed and configured to allow you to send the certificate request.
- Root access on the NetServer host.

System Requirements

Host Requirements

A host running NetServer 6.0.0 must meet these requirements:

- Pentium II processor (or equivalent RISC processor)
- 512 MB RAM
- 128 MB free RAM, 256 MB recommended
- 256 MB permanent disk space, 500 MB recommended

Installation Requirements

The NetServer installation process requires these system resources:

- 60 MB temporary disk space
- SMTP services for transmitting the certificate request
- Enrollment requests can be sent using FTP if SMTP services are not available.

Supported Platforms

NetServer 6.0.0 is supported on following platforms:

- Linux RHEL 5.5 32-Bit
- Linux RHEL 5.5 64-Bit
- CentOS 5.7 32-Bit
- CentOS 5.7 64-Bit
- Ubuntu 12.04.2 LTS 32-Bit
- Ubuntu 12.04.2 LTS 64-Bit
- Solaris 10 Sparc 32-Bit
- Solaris 10 Sparc 64-Bit

Interoperable RADIUS Servers

The list of RADIUS servers with which NetServer is interoperable includes, but is not limited to:

- FreeRADIUS (recommended)
- RADIATOR (recommended)
- Cistron RADIUS
- DTC RADIUS, v2.02 and later
- Interlink Networks Advanced Server (AAA)
- FUNK Steel-Belted RADIUS v3.0 and later
- Ascend Access Control (Extended RADIUS)
- Ascend RADIUS 960112, 970224
- Vircom RADIUS
- Navis RADIUS
- Merit (Enterprise Editions only)

Additional Operational Requirements

Additional operational requirements include:

- Connectivity to a primary RADIUS capable of proxying authentication and accounting packets.
- Domain Name Server (DNS) installed and configured to work with the NetServer host.
- Connectivity to the iPass Transaction Servers. The TCP/IP protocol is required to support the SSL-encrypted connection between the NetServer and the iPass Transaction Centers.
- Other processes, such as a firewall or authentication server, can be run on the platform concurrently with NetServer.

Firewall Rules

If NetServer 6.0.0 is installed behind a firewall or other network address translation device, you must enable the firewall rules shown in the following table. Notes at the end of the table give more information.

Purpose	Inbound	Source IP(s)	Destination IP(s)	IP	Port	Protocol
iPass Transaction Center auth-apac.ipass.com(Hong Kong,CN)			x	216.239.98.126	9101	TCP/IP
iPass Transaction Center auth-sjc.ipass.com(San Jose,CA)			x	216.239.108.126	9101	TCP/IP
iPass Transaction Center auth7.ipass.com(Atlanta,US)			x	216.239.111.126	9101	TCP/IP
iPass Transaction Center auth8.ipass.com(London,UK)			x	216.239.105.126	9101	TCP/IP

Purpose	Inbound	Outbound	Source IP(s)	Destination	Port	Protocol	Notes
Monitoring		x		216.239.99.200	1984	TCP/IP	
Monitoring	x		216.239.99.200		1984	ICMP(ping)	
Monitoring		x		216.239.100.200	1984	TCP/IP	
Monitoring	x		216.239.100.200		1984	ICMP(ping)	
Configuration Upload Server		x		216.239.111.209 216.239.111.200	9101	TCP/IP	NetServer sends its configuration file on a regular basis to the Configuration Upload Servers.
Software Update Server		x		216.239.99.209 216.239.99.200	9101	TCP/IP	NetServer periodically checks for software updates on Update Server.

SSH access for troubleshooting and routine maintenance.	x		216.239.97.227		22	TCP/IP	SSH access from the iPass Operations Center should be allowed for troubleshooting and routine maintenance.

Supported RADIUS Attributes

NetServer 6.0.0 supports the following RADIUS attributes:

- All attributes form RFC 2865 and 2866
- From RFC 2869: EAP-Message, Message-Authenticator, NAS-Port-Id
- From RFC 4372: Chargeable-User-Identity (CUI)
- From RFC 2546: ms-mppe-send-key, ms-mppe-recv-key

Graceful Forwarding: NetServer authentication and accounting will drop attributes that are not listed in RFC 2865 and 2866, but packets are still forwarded.

NetServer Default Ports

- SSL port=11811
- Authorization port=11812(NetServer uses a different port than RADIUS)
- Accounting port=11813
- Proxy authorization port=11817
- Proxy accounting port=11818

Installation Process

The installation process consists of downloading the installation file and then installing the software.

Downloading

You will need to download NetServer installation file from our secure FTP site. Contact your iPass installation engineer for your FTP username and password.

To download the NetServer installation file:

1. At a command line, type: ftp ftp.ipass.com
2. At the username prompt, type your FTP username.

3. At the password prompt, type your FTP password.
4. Type: cd NS/6.0.0
5. Type: bin
6. Type: get <correct version for your OS>
7. When the download is complete, type: bye.

Directory names and filenames are case-sensitive.

Installing the Software

This guide uses the term <NS_Home> for the NetServer 6.0.0 installation directory. The default is /usr/ipass/netserver/current_version.

To install the NetServer 6.0.0 directories:

1. The iPass Products directory /usr/ipass should be created and owned by root or the appropriate “ipass” service account with sudoer permission. If your partitioning scheme does not provide enough space for /usr/ipass, create a symbolic link for the /usr/ipass directory to the intended installation volume.
2. Copy the downloaded NS 6.0.0 installation file “netserver_6.0.0_<platform>.tar.gz” under '/usr/ipass/',where <platform> is the platform of your NetServer.
3. Type: cd /usr/ipass/
4. Type: gunzip netserver_6.0.0_<platform>.tar.gz to uncompress the installation file.
5. Type: tar -xvf netserver_6.0.0_<platform>.tar. By default, this will create a hierarchy in ”/usr/ipass/netserver/6.0.0” with all the necessary directories and files. In order for NetServer to run correctly,you must keep the file hierarchy as it is installed.
6. Run ./create_link script from the directory path ”/usr/ipass/netserver/6.0.0/.scripts” to create a softlink /usr/ipass/netserver/current_version → /usr/ipass/netserver/6.0.0
7. Type: ipassconfig.csh -conf under /usr/ipass/netserver/current_version/bin to generate ipassNS.properties file. For detailed information regarding configuration, refer to Configuration section. Once configuration is complete, your NetServer is ready to be started.
8. Run init.sh script under /usr/ipass/netserver/current_vesrion/bin directory to create RC scripts.
9. Review the End User License Agreement from the file ”<NS_Home>/license.txt”.

Migration Tool

Migrating from NetServer 5.x to NetServer 6.0.0

If you are upgrading to NetServer 6.0.0 from version 5.x, the Migration Tool has to run manually post installation process. The Migration Tool will convert your old configuration file into the new ipassNS.properties adding newly added properties in 6.0.0, and copy certificates and keys from the old installation into KeyStores.

Manual Migration

- Untar NetServer 6.0.0 build under /usr/ipass/ to create a hierarchy /usr/ipass/netserver/6.0.0

1. *Migrating from NetServer 5.x to 6.0.0*: Type ./ns_migration_tool.csh under /usr/ipass/netserver/6.0.0/bin

For example, cd /usr/ipass/netserver/6.0.0/bin and run : ./ns_migration_tool.csh. It will prompt you for the path to migrate files from. Enter NS5.x path /usr/ipass/netserver/5.x. It should migrate ipassNS.properties file,certs and keys from 5.x version to 6.0.0 and it will add new attributes to ipassNS.properties as below:

- 'strip' flag is added for Routing Realms and for realms 'ipass' and 'ipasv' it is set to 'yes' by default and for other realms it is set to 'no' by default.
- 'AdminKeyStoreProperty' is added to ipassNS.properties.
- All .pem keystore files are migrated to .keystore extension

2. Once migration is done run create_link.sh script from the path ”/usr/ipass/netserver/6.0.0/.scripts” to create soft link(It will create soft link under ”/usr/ipass/netserver/current_version”).

3. Run init.sh script from /usr/ipass/netserver/current_version/bin to create RC scripts.

The NetServerd Script

The script NetServerd is not included in the Migration Tool process, so command line options it contains will not be carried over to the new version of NetServer. This may trigger the following issues:

Non-Default Ports: The NetServer 6.0.0 Migration Tool assumes that your NetServer runs on the default port of 11811. If this is not the case, after you run the Migration Tool, you will need to edit the following attributes in the ipassNS.properties file:

- Listener1=Type=RADIUS,Port=<port number>
- Listener2=Type=RADIUSProxy,Port=<port number>
- **Dual-Homed Hosts**: If NetServer 5.x runs on a dual-homed host, the Migration Tool may not bind NetServer to the correct IP address. You will need to check that the ipassNS.properties file reflects your correct IP address.
- **Port Settings**: The migration tool will automatically migrate your previous port settings from NetServer 5.x to 6.0.0.

Rollback Procedure

If you need to roll back your NetServer 6.0.0 installation to a previous 5.x version, follow the appropriate procedures listed here.

These instructions assume that NetServer 5.x is installed in /usr/ipass/netserver/5.x, and NetServer 6.0.0 is installed in /usr/ipass/netserver/current_version.

To rollback NetServer 6.0.0:

1. If necessary, stop NetServer 6.0.0 as follows:
   - Type: cd /usr/ipass/netserver/current_version/bin
   - Type: ./netserverd stop
   - Check if the process stopped by typing: ps -auxwww | grep netserver
   - If the process did not die, execute: ./netserverd kill
   - Verify that the process stopped by typing: ps -auxwww | grep netserver
2. Change the softlink file /usr/ipass/netserver/current_version to point back to the previous NetServer directory /usr/ipass/netserver/<NS Version>, as follows:
   - cd to /usr/ipass/netserver/
   - rm current_version
   - ln –s /usr/ipass/netserver/5.x current_version
3. Start the old NetServer:
   - cd to /usr/ipass/netserver/<NS Version>/bin
   - run ./netserverd start

Uninstalling NetServer 6.0.0

- Type: cd /usr/ipass/netserver
- Type: rm –rf 6.0.0

NetServer Binding

To bind to a local IP for outgoing requests to the Transaction Servers, you need to configure the LocalIpAddress attribute of your IpassServers property:

To view iPass Transaction Server information, type: <NS_HOME>/bin>ipassconfig.csh -help IpassServer

Sample format of IpassServer:

IpassServer1 = name11=value11,name12=value12,...

IpassServer2 = name21=value21,name

See the Property Glossary for more information on configuring this value.

Go to: **Other Product Documents > NetServer Admin Guide**

netserver, installation, requirements

Configuration

There are two ways to configure your network architecture to allow the NetServer to route iPass access requests. These configurations vary depending upon where the NetServer is installed in relation to your NAS and AAA servers.

The following details are available:

Configuration Types

NetServer Behind the AAA Server

The most common configuration, and the one iPass recommends, places the NetServer behind the provider's entire authentication system. In this scenario, all incoming authentication requests are received by the NAS and forwarded to the RADIUS AAA server. The RADIUS server performs the primary sorting and routing functions, separating iPass users from the provider’s other users.

When iPass requests are received, the RADIUS server will forward the packets to the NetServer, which will then forward them to one of the iPass Transaction Centers. The RADIUS server will recognize all other packets as provider requests and authenticate the users accordingly.

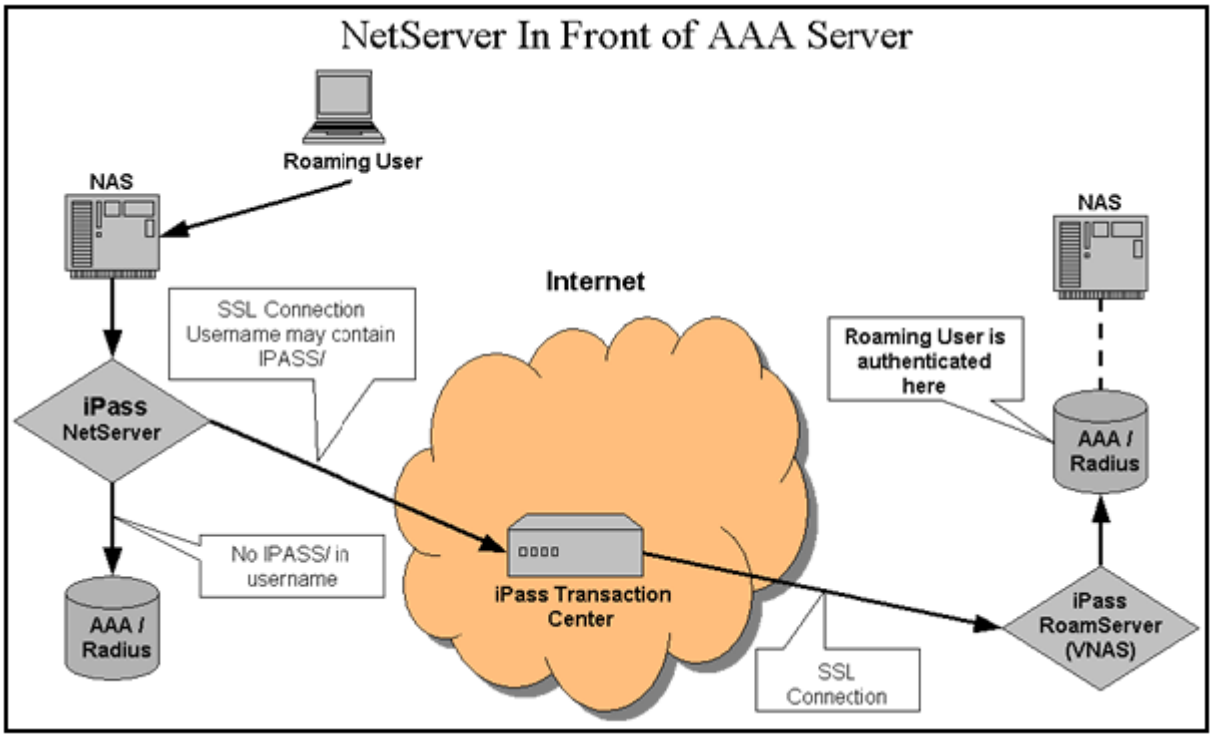
With NetServer behind the AAA, in the rare event of NetServer failure, normal AAA authentication procedures are not impacted. Also, if there is a problem with accounting records received by the iPass Transaction Center, the provider can submit copies from the AAA server to resolve discrepancies.

This solution can only be implemented if the existing RADIUS server supports proxy.

NetServer In Front of the AAA Server

In this scenario, the NetServer will perform the primary sorting and routing functions, separating local users from iPass customers. The NetServer will scan the realm of each packet, and route all requests

with the IPASS/ prefix to an iPass Transaction Center. All other packets will be routed to the local AAA server for authentication.



NAS Configuration

To support this type of network architecture, you must reconfigure the NAS to proxy all access requests to the NetServer rather than the AAA server. To do so, you will need to use the NAS configuration utility to allow the NAS to forward all authentication and accounting requests to the NetServer's IP address, port numbers, and the shared secret listed for the NAS in the ipassNS.properties file.

For additional information about configuring your NAS, please refer to the documentation included with the software or contact the manufacturer for assistance.

ipassNS.properties File

The main NetServer configuration file is called ipassNS.properties. By setting properties in the file, you can enable or disable NetServer functions. (Enabling some features might involve setting more than one property.)

NetServer will periodically upload its encrypted ipassNS.properties to an upload server, including at startup. This information will be used for diagnostic and troubleshooting purposes across the iPass network.

An example of the ipassNS.properties file is shown in Appendix 1.

- **Viewing Properties**
  - *To view any property value*, run: ipassconfig.csh -get <property name>
  - *To view all property values*, run: ipassconfig.csh -listall

- **Editing Properties**- You can edit the file and add, change, or delete properties in several ways:
  - Run **ipassconfig.csh -conf** in your <NS_Home>/bin directory. This is the recommended method.
  - To set a specific property value, run: **ipassconfig.csh -set <property name> <value>**
  - You can also use a text editor. However, we strongly recommend use of the ipassconfig.csh script, when possible, to ensure correct naming and formatting of property names and values.
  - To set a new property value in a text editor, open the file and type in the name and value of a new property. If a text editor is used, properties should be set by entering: **<property name>=<value>**

You will need to reload the properties file, or restart the NetServer, in order for your edits to go into effect.

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

For information on particular properties, see the Property Glossary.

Configuration Procedure

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

Important Note

You will be running the **<NS_Home>/bin,run ipassconfig.csh -conf** script.

Configuration Checklist

NetServer configuration consists of the tasks in this checklist. **Page** indicates the page of this document where the procedure is described in more detail.

Task	Page
1. Set initial configuration by running ipassconfig.csh -conf	16
2. Enroll an iPass Transaction (ns) Certificate for the NetServer.	17
3. If necessary, configure your RADIUS server for iPass traffic.	36
4. Test the installation.	18

Initial Configuration

Initial configuration is done by running the ipassconfig.csh script, which sets many of the properties in your ipassNS.properties file.

To initially configure NetServer:

In <NS_Home>/bin,run ipassconfig.csh -conf. 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.

1. Time, Date and Timezone 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.

2. Customer ID: (Default Value=1) Enter your customer ID, supplied by iPass. This is the same ID number used on your iPass Web site login.

3. 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. See *Trace Log File*on page 30 for more information.

4. Authorization Port: (Default Value=11812) Enter the NetServer authorization port. iPass recommends you use port 11812.

5. Proxy Listening Port: (Default Value=11817) Enter the NetServer proxy listening port. iPass recommends you use port 11817.

6. NetServer requires a keystore for the SSL encryption (either for NS to TS communication or for EAP mode). The different types of keystores for NetServer include:

- **EAP Keystore**: Production Grade Keystore (<NETSERVER_HOME>/certs/eapserver.keystore) signed by Thwate, bundled as part of NetServer installation for EAP-TTLS authentication over WPA. This keystore is required only when EAPMode is enabled on the Netserver. It is not required to generate a new KeyStore for this type. When the tool prompts the following text, please press “K” to keep this configuration. Verify the similar console output as given below:

```
KeyStore1=KeyStoreType=eap,KeyStorePath=$ipass.server.home/certs/eapserver.keystore,KeyPassword=UfGjld0YWEUjEIZUnNvIsA==,KeyStorePassword=UfGjld0YWEUjEIZUnNvIsA==

Would you like to: (E)dit or (K)eep the keystore settings: [K] K
```

- **NS Keystore**: iPass transaction system certificates (ns#.keystore). This Keystore contains a trusted CA and certificate keypair signed by iPass used for NS to TS SSL encryption. The tool will prompt with following information required to generate the NS type keystore. Verify the similar console output as given below:

```
KeyStore2 does not exists.

Plesae Edit the keystore settings:

KeyStoreType=ns
```

```
Please enter the
KeyStorePath:[/usr/ipass/netserver/current_version/certs/ns1.keystore]
/usr/ipass/netserver/current_version/certs/ns1.keystore

Please enter the KeyAlias:[ns] ns

Please enter the CertAlias:[ipassca] ipassca

Please enter the Salt:[iPassNS] iPassNS

Please enter the KeyPassword:[changeme] changeme

Please enter the KeyStorePassword:[changeme] changeme

New Server 2 settings:

KeyStoreType=ns,KeyStorePath=/usr/ipass/netserver/current_version/certs/ns1.
keystore,KeyAlias=ns,CertAlias=ipassca,Salt=iPassNS,KeyPassword=<HASHED
PASSWORD>,KeyStorePassword=<HASHED PASSWORD>

Attempting to set Property (KeyStore2) with value

KeyStoreType=ns,KeyStorePath=/usr/ipass/netserver/current_version/certs/ns1.
keystore,KeyAlias=ns,CertAlias=ipassca,Salt=iPassNS,KeyPassword=<HASHED
PASSWORD>,KeyStorePassword=<HASHED PASSWORD>
```

7. To complete installation you will need to obtain a signed certificate from iPass. For more details, please visit our **iPass Certificate Enrollment** help page.

```
Upon completing the steps, tool will prompt for the CSR generation process.
It is necessary to enter the appropriate values that are associated with the
CSR property, otherwise make use of the default value,Verify the similar
console output as given below :

Would you like to generate the Private key and Certificate Signing Request
for KeyStore2:[yes/no]yes

Key Size[2048]2048

Country code [US]US

State or Province Name [Some-State]Some-State

City or town [Some-City]Some-City

Company Organization name [Some-Organization]

Enter the public IP address by which this system is known to the outside
world. If this system has multiple IP addresses, please specify the address
which will appear as the source address for connections to the iPass
authentication servers.  This address will be used for authenticationand
```

```
authorization purposes by the iPass software.:[192.168.232.154]

Enter the fully qualified domain name by which this system is known to the
outside world.  If this system has multiple domain names, please specify the
name which is to be used for iPass purposes. This domain name will be used
for authentication and authorization purposes by the iPass
software.:[bld-qa-net12]

Enter your e-mail address:[user@domain.com]

On pressing Enter it generates keystore for SSL connections as :

Will be generating keystore for SSL Connections

Press enter to continue...

Generated Certificate request successfully, and saved it to:

/usr/ipass/netserver/current_version/certs/mail_cert_req2.data
```

7a. Import the contents of the certificate to KeyStore [/usr/ipass/netserver/current_version/certs/ns1.keystore] using the following command:

```
/usr/ipass/netserver/current_version/bin/ipassconfig.csh -import_cert
<KeyStoreProperty> "<Signed_Certificate_File_Path>"
```

8. Transaction Servers: (Default Value=no). If you wish to configure your NetServer to communicate with the iPass transaction servers, enter **yes**. You will need to enter each server's IP address and other relevant configuration parameters.

9. RADIUS Clients: (Default Value=yes). If you wish to configure your NetServer to communicate with your RADIUS clients, enter **yes**. You will need to enter each server's IP address and other relevant configuration parameters.

List your NS keystore certificate

To list your certificate, in <NS_Home>/bin, run the script list_NS_keystore.csh.Usage is :

- ./list_NS_keystore.csh
- ./list_NS_keystore.csh <KeyStoreFilePath>
- ./list_NS_keystore.csh <KeyStoreFilePath> <Password>

For example : ./list_NS_keystore.csh ../certs/ns1.keystore changeme

It will list the certificate chain entries, **valid from** and **valid to dates**, **Issuer**, **Owner** and imported **ipassCA** details. KeyStore certificate entries in ipassNS.properties should be valid/non-expired in order to start NetServer.

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 ipassNS.properties, so you won't have to re-enter those values.

For example, two months after you install NetServer, you decide to add a secondary authorization server. You would run ipassconfig.csh -conf, 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.

EAP-TTLS Authentication Mode

The Netserver comes equipped with EAP-TTLS authentication using its on-board eap.keystore and 2048-bit RSA Server Certificate issued by Thawte to netserver.ipass.com on behalf of iPass Inc. This authentication mode runs in parallel with standard RADIUS authentication at the listening port.

Using a single iPass configuration profile for TTLS/X globally, the Netserver assists in tunneling the iPass EAP-ID's **second-phase authentication** (GTC, PAP, CHAP, MSCHAPv2, EAP-TLS, EAP-SIM) over the iPass Federated AAA fabric so that it emerges from the iPass Roamserver located at the Home AAA. The tunneled authentication must contain an @domain in the credential's User-Name so that it can be routed correctly. If the Home AAA Accepts the phase two authentication, then the session is granted by the Netserver.

This TTLS Authentication Method not only simplifies global provisioning and provides additional security all-around by allowing for advanced authentication at the Home AAA, but it also provide a highly secure “wrapper” around Legacy-based standard RADIUS authentication methods for customers that do not support advanced authentication at their AAA.

EAP-RADIUS can be configured to pass-through the Netserver to iPass by protocol number. By default only TTLS tunneled auth is enabled.

Note that in order to pass EAP-enabled RADIUS traffic over iPass without using the Netserver's native EAP-TTLS tunneling mode will require additional provisioning and policy by iPass.

To use Anonymous identities and automatic Chargeable-User-Identity reversal when transacting over EAP-enabled networks, a set of pre-configured authorized ID Tokens are provided, e.g. anonymous@ipass.com. Otherwise - to prevent Tunnel Fraud - the User-Name of the tunneled phase two authentication must be reflected somewhere within the EAP-Identity during the initial Access-Request, e.g IPASS/**user@home.com**@ipass.

Please consult with your iPass representative about advanced network implementations using both standard RADIUS and EAP Authentication Methods facilitated by an iPass Netserver.

Configuration Testing

Once you have finished configuring your NAS or RADIUS to route iPass access requests to your NetServer, you must test your network to ensure proper functioning. Before configuration testing, ensure that you have set up the NetServer as a client of your RADIUS.

There are two configuration tests you need to perform:

- checkipass.csh
- a connectivity test using iPassConnect/OpenMobile

checkipass in SSL Mode

The checkipass test verifies that the NetServer can communicate with the iPass Transaction Server. When ran in SSL Mode, the request passes through the corporate firewall to the iPass Transaction Server.

- You will need to use a valid user name and password for the host on which the NetServer is installed.
- Optimally, in order to run the checkipass test with no realm from the NetServer to the AAA, the AAA server should be configured with the NOREALM option in the routing realm. For example, RoutingRealm1=realm=NOREALM, AuthServer=ProxyAuthServer1, AcctServer=ProxyAcctServer1
- ProxyAuthServer1 should be configured for AAA server.
- To run checkipass in SSL mode, in <NS_Home>/test, type: ./checkipass.csh –proto SSLPost [options] -u <userid>

[options]

Your options are:

-p <password>	Specifies the user password. Otherwise, you will be prompted for it.
-host <hostname or IP>	Host Name or IP address to send request to.
-port <port number>	Port number of the destination host. for RADIUS, this would be the authentication port.
-type <request type>	Request type. Default is normal. Choices are auth, acct, start, stop, all, normal.
-keystore <KeyStoreProperty>	KeyStore name to be used to ssl connection.
-timeout <timeout>	Timeout, in milliseconds, to wait for a reply. Default is 60000 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.
-secret <secret>	The RADIUS shared secret.
-proto <proto>	Protocol of request. Choices are SSLPost or RADIUS.

-attr<name=value>	Name=Value consists of pairs of attributes to add to the packet.
	Example: -attr namel=value1 -attr name2=value2. rs_ip_address=<ip address> -attr rs_port=<port>
	Use this to specify which RS should handle the request. record_stats=y
	Use this to get back statistics on connection times. vendor_specific <vendorID:vendorTYPE:value>
	Use this to test Vendor-Specific functionality, where vendorID is a positive number and vendorTYPE is a number between 0 and 255. nas_ip=<x.x.x.x>
	Use this to test NAS-IP-Address related poilcy. framed_ip_address=<x.x.x.x>
	Use this to test Framed_IP-Address related policy. location_id=<location id>
	Use this to test location_id related policy. called_number=<called station id>
	Use this to test Called-Station_Id related policy. called_number=<phone number>
	Use this to test Called-Station_id related policy.
-show_radius_attrs	Shows the supported list of RADIUS attributes.
-interactive	Run the tool in interactive mode.
-help	Show the help/usage of the tool.

The test output will show the status of the checkipass test = Accept or Reject. If status = Accept then the NetServer 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 certificate, dates and readability of your certificate by running the tool list_NS_keystore.csh in your <NS_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 NetServer is running on port 9101.
- Invalid shared secret. Verify that your RADIUS shared secret is entered properly.

Test Cycle

Run the checkipass test once for each iPass Transaction Server, using the -host option and changing the IP address each time to reflect each Transaction Center IP.

Connectivity Test Using iPass Client

This test will verify that iPass users are able to connect to the iPass network through your access points.

Shortly after your installation and configuration is complete, your iPass NetServer Installation Engineer will send you a customized version of the iPassConnect / OpenMobile client software for testing purposes, along with your test username and password.

To run the connectivity test:

1. Using the iPassConnect/OpenMobile client, and your test username and password, connect to each of your access points.
2. Repeat this test at least 10 times for each access point.

checkipass in RADIUS Mode

This optional test verifies NetServer connectivity across your network. When run in RADIUS Mode, the checkipass request passes from the NetServer, through the corporate firewall, to the iPass Transaction Server, as well as to the AAA server and proxy server.

To run checkipass in RADIUS mode, in <NS_Home>/test, type: ./checkipass.csh –proto RADIUS [options] -u <userid>, where [options] are described on page 19.

Next Steps

After testing, the NetServer installation and configuration process is complete. Your network should now be configured to allow iPass traffic to be routed to the iPass Transaction Centers. Once your initial Dial-up testing is complete, the iPass Network Quality department will verify the quality of your network before pushing your access points to the iPass roaming users around the world. This final testing may take up to one month, after which you will enjoy the many benefits of being an iPass provider.

Go to: **Other Product Documents** > **NetServer Admin Guide**

netserver

iPass Certificate Enrollment

To complete installation you will need to obtain a signed certificate from iPass. If this is your first NetServer installation then please contact your Sales Representative for details. If you are already an iPass Customer then please request this service through your usual iPass Support channel.

To add this server to the iPass Network you will also need to supply the following information:

- **Public IP address**
- **Operating system name**
- **Operating system version**
- **Operating system kernel and service pack**
- **Firewall products in use**
- **Authentication protocol** (e.g. **RADIUS**)
- **System host name**

If you want to generate more ns type keystore, you can repeat the above process,or can skip the step.Next step is to import the certificate to keystore(ns1.keystore) as:

- If SMTP services are available on this server or another computer, you should email the contents of <NS_Home>/certs/mail_cert_req.data as a text file attachment to the network analyst responsible for your deployment or the email alias ops-so@ipass.com.
- If SMTP services are not available on this server, you can exchange certificates in real time with an iPass technician using FTP. Contact your iPass installation engineer to arrange this exchange.

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

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

This process may take up to 48 hours. If you need the certificate immediately, please contact iPass Technical Support.

Once you have the signed certificate, place the same under **/usr/ipass/netserver/current_version/certs** folder and you have to import the same to the respective keystore to complete the process. Use the ipassconfig.csh to import the same, type the following command to import the signed certificate:

```
/usr/ipass/netserver/current_vesrion/bin/ipassconfig.csh -import_cert
<KeyStoreProperty>
"<Signed_Certificate_File_Path>"
```

For instance, CSR generated for the KeyStore2 and to import the signed certificate mail_cert_req2.crt, do the following:

```
./ipassconfig.csh -import_cert KeyStore2 "<NETSERVER_HOME>/certs/
mail_cert_req2.crt"

Importing a Trusted certificate from keystore

[/usr/ipass/netserver/current_version/certs/ipassCA.keystore] to keystore
[/usr/ipass/netserver/current_version/certs/ns1.keystore]

Importing a signed primary certificate
[/usr/ipass/netserver/current_version/certs/mail_cert_req2.crt] to a Java
keystore [/usr/ipass/netserver/current_version/certs/ns1.keystore]
```

Go to: **NetServer Admin Guide** > **Configuration**

ipassNS.properties

The ipassNS.properties file allows configuration of NetServer features. By setting properties in the file, you can enable important NetServer functions. Enabling some features may involve setting more than one property.

NetServer will periodically upload its encrypted ipassNS.properties to an upload server, including at startup. This information will be used for diagnostic and troubleshooting purposes across the iPass network.

Property Help

You can obtain help on any property by running ipassconfig.csh, found in your <NS_Home>/bin directory.

To list all server properties: ipassconfig -listall

To describe usage of a property: ipassconfig -help <property name>

Property Glossary

This glossary defines all properties found in ipassNS.properties, including configurable parameters for each property.

Property	Description
AcctLogBackupType	AcctLogBackupType=<backupType> where <backupType> is either MultipleWithTimestamp or SingleBackup. The default is MultipleWithTimestamp. AcctLogBackupType sets the accounting log's backup file name when rotation is to be performed on local accounting files.
AcctLogRotationDays	AcctLogRotationDays=<days>. Valid range is: 1 to 30 days. the default is 7 days. AcctLogRoatationDays control how often the local accounting file is rotated.
AcctLogRotationMaxSize	AcctLogRotationMaxSize=<max size>. Minimum value is 100 kbytes. Maximum value is 20000 kbytes. The default is 10000 kbytes. AcctLogRotationMaxSize limits how large (in kbytes) the local accounting file can get before it is rotated.
AcctLogRotationType	AcctLogRotationType=<rotationType> Where <rotationType> is either FileSize or NumberOfDays. AcctLogRotationType sets the type of rotation to be performed on the local accounting files. The default is FileSize.
AdminKeystoreProperty	AdminKeyStoreProperty=<KeyStoreProperty> determines which KeyStore will be used for administrative purposes. This property must be specified.
AllowAcctUpdate	When AllowAcctUpdate is set to YES, this server will allow accounting Interim-Update requests to be forwarded to the iPass network. The default value is set to NO.
AppendNasPortType	Determines if nas_port_type needed to be appended to the called_station_id. If set to true, called_station_id is modified to called_station_id:nas_port_type in both the Auth and Acct packets forwarded to iPass. Default is set to false.

AscendDataFilter	AscendDataFilter1=<valid string for ascend-data-filter>. This is used as an anti-spam feature for some providers and will block the email port (25) at the provider.
	If the AAA server does not send it to us, we will use the AscendDataFilter(s) specified to send back in the authorization accept packet.
	An example entry is: AscendDAtaFilter1=ip in forward tcp est. AcsendDataFilter2=ip in forward dstip xxx.xxx.xxx.xxx/yy. AscendDataFilter3=ip in drop tcp dstport=25. AscendDataFilter4=ip in forward.
	The string ip in drop tcp dstport=25 is a mandatory AscendDataFilter attribute. When no AscendDataFilter is configured, this feature is disabled.
AuthCacheDays	AuthCacheDays=<# of days> This attribute determines the maximum amount of days an authentication reply is cached by the NetServer. Valid range is 1 to 7 days. The default value of this property is to set to 7 days.
AuthCacheEnabled	AuthCacheEnabled=yes/no. Determines if the caching authentication requests is enabled. Default is set to YES.
AuthCacheSize	AuthCacheSize=<number of users> This attribute determines the maximum amount of successful user authentication replies cached by the NetServer. Valid range is 60 to 10000 users. The default value of this property is set to 500 users. If an odd value is specified, then the allowed cache size is the next even number.
AutoUpdate	AutoUpdate=yes/no. Determines if automatic software update is enabled. Default is set to FALSE.
AutoUpload	AutoUpload=TRUE/FALSE. Determines if automatic file upload is enabled. Default is set to TRUE.
CollectStatistics	CollectStatistics=yes/no. Determines if statistics should be collected. Default is set to TRUE.
CustomerId	CustomerId=<iPass Code>. This is the same number as your iPass portal customer ID. Default value=1.
CuiEnable	CuiEnable=yes/no or true/false. If EapNaiCheck is false, a supplier must support attribute 89 in all Radius Accounting. These settings enable reflection of Chargeable-User-Identity Attribute 89 in all Access-Accept. The value of the attribute is the accepted userid negotiated with iPass Transaction Centers unless a CUI is returned by the Home AAA.Default setting is: TRUE.
CuiHashEnable	CuiHashEnable=yes/no or true/false. CuiHasEnable encodes the value of the CUI returned in an Access-Accept to create a unique and anonymous identity hash of the user portion of the NAI or of complete NAI. This hash is reversable by iPass for billing correlation. Default setting is: false
CuiHashingStrategy	CuiHashingStrategy=<cuiHahingStrategy> Where <cuiHahingStrategy> is either NAI or USER_NAME.The default is USER_NAME.CuiHashingStrategy sets the type of hashing strategy to be performed on CUI hashing.That is, either the complete NAI or just the USER_NAME in an NAI to be hashed.
CuiAcctUserReplace	Provides CuiAcctUserReplace information. Sample format of the entries: CuiAcctUserReplace1 = name11=value11,name12=value12,... CuiAcctUserReplace2 = name21=value21,name22=value22,... Below are the list of various CuiAcctUserReplace attributes:Token : CuiAcctUserReplace retrieves the CUI from all Accounting messages and replaces the Anonymous EAP Identity token used for routing the transaction in the User-Name with the value of the CUI returned in the Access-Accept Decode : If CuiHashEnable is yes, CuiAcctUserReplace can optionally leave the User-Name encoded for transmission to iPass for correlation by the Home AAA without CUI.
DebugLevel	DebugLevel=<level>. Debug level determines if debug and error messages are logged to the trace file. The following levels are supported.
	Debug Level 0- Only severe messages are logged.
	Debug level 1-Error messages are logged.
	Debug level 2-Error and Debug messages are logged.
	Debug level 3-Error, Debug, and Packet parsing information is logged.
	Debug level 4-Error, Debug, Packet parsing, and Packet dumping is logged.
	Debug level 5-Detailed Packet and debug information is logged.
	The default value for this property is 0. Production servers should normally run with debug level 0.
DupFilterCleanupDelay	DupFilterCleanupDelay=<# of seconds>. This attribute determines the amount of time in seconds to continue duplicate filtering a completed authentication requests. Valid range is 0 to 60 seconds. The default value of this property is set to 2 seconds.
DupFilterTimeToLive	DupFilterTimeToLive=<# of seconds>. This attribute determines the maximum amount of time in seconds to cache all attempted user authentication requests. Valid range is 5 to 60 seconds. The default value of this property is set to 30 seconds.

DuplicateFilterByUid	DuplicateFilterByUid=yes/no. When enabled, duplicate detection will be done solely based on the user ID. When disabled, duplicate detection will be based on the source IP Adress, source port, and Identifier of the RADIUS packet. Default is set to: NO.
EapMode	EapMode=yes/no or true/false. Determines if the NetServer will do early-termination of EAP_TTLS/PAP requests. All other EAP types will be blocked unless otherwise conifigured to do so. Default setting is:true.
EapNaiCheck	EapNaiCheck=yes/no or true/false. Determines if the NetServer will check that the inner NAI contained the Outer NAI of a tunnled request, prior to forwarding to iPass. This only applies to EAP Early-Terminated tunnled protocols. Default setting is: true.
EapNotification	EapNotification=yes/no or true/false. Determines if the NetServer will send back the Reply-Message(s) in EAP-Notification Requests prior to sending back the final RadiusAccess-Accept/Access-Reject. Default setting is: NO.
EapNotificationFilter	EapNotificationFilter1=<Reply-Message prefix string>.
	Expected format is: EapNotificationFilter1=FilterPrefix=<filter string>, KeepPrefix=<yes/no>. This feature is used in conjunction with the EapNotification feature. It is used to filter which Reply-Message(s) can get sent back to EAP-Notifications. It will check if any Reply-Messages begin with the given FilterPrefix string.
	FilterPrefix: The string to match at the beginning of the Reply-Message. It is case intensive.
	KeepPrefix: Whether to keep that prefix attached to the Reply-Message when sending back as an EAP-Notification.
	An example entry is: EapNotificationFilter1= FilterPrefix="Location=",KeepPrefix=YES EapNotificationfilter2= FilterPrefix=iPassTAG,KeepPrefix=NO.
	When no EapNotificationFilter is configured, then nothing is filtered/blocked. This means the server will send back all Reply-Message(s) as EAP-Notifications, as long as EapNotification has been enabled.
EapPassThroughAllow	EapPassThroughAllow=<EAP Protocol Type>. Determines if a NetServer in EAP Mode (early-termination) will allow the mediated pass-through of other EAP protocols end-to-end. The <EAP Protocol Type> can be the either one of the keywords all, or nothing, or a list of EAP type protocol numbers separated by commas. When nothing is configured, then nothing is allowed to pass. Default setting is: nothing.
EapPassThroughDeny	EapPassThroughDeny=<EAP Protocol Type>. Determines is a NetServer in EAP Mode (early-termination) will deny the meditated pass-through of certain EAP protocols end-to-end. The <EAP Protocol Type> can be either the keyword. When nothing is configured, then nothing is explicitly denied passage. Default setting is:nothing.
EapEarlyTerminate	EapEarlyTerminate=<EAP Protocol Type> Determines if a NetServer in EAP Mode (early-termination) will allow other EAP protocols (other than TTLS) to early-terminate at this server The <EAP Protocol Type> is a list of EAP type protocol numbers separated by commas Valid values are: 4 (EAP-MD5) and 25 (EAP-PEAP) When nothing is configured, then only TTLS will early-terminate Default setting is: null.
EnableEquantDna	EnableEquantDna=yes/no. Determines if the NetServer should send the first 4 bytes of Calling-Station-Id as Equant-DNA to the iPass Transaction Center. Default is set to false.
HeartBeatInterval	HeartBeatInterval=<number of minutes>. This entry determines the time interval between heartbeat messages. This is an advanced setting. The server may not function properly if this value is set incorrectly. Default value for this property is set to 15 minutes.
HeartBeatMessage	HeartBeatMessage=yes/no. This entry determines if the heartbeat is turned on or off. This is an advanced setting. The server may not function properly if the value is set incorrectly. Default value for this property is set to no (heartbeat messages are turned off)

IpassServer	Provides iPass Transaction Server information. Sample format of the entries: IpassServer1=name11=value11,name12=value12,... IpassServer2=name21=value21,name22=value22,...
	IpassServer attributes:
	IpAddress: The iPass Transaction Server's hostname or IP address.
	LocalIpAdress: The Local IP address to bind the socket to. (Optional)
	Port: The server's port number.
	ConnSharing: This is used for persistent SSL connection. If this is set to 1, then the connection is shared between requests. A value of 0 means the feature is disabled. The default value is 0.
	SslSessionExpTime: The maximum duration of a persistent SSL connection. Valid range is 10 to 480 minutes. The default value is 10 minutes.
	FailureThreshold: Once the failure count exceeds the Failure Threshold, the server is removed form the list. The default value is 4.
	InititalPingInterval: A thread will be launched to ping a failed Transaction Server. The first ping is sent out according to the InitialPingInterval. The default value is 60 seconds.
	PingBackOffFactor: If there is no response, then the next ping is sent out according to the InitialPingInterval multiplied by the PingBackOffFactor. The default value is 2. FinalPingInterval: This process is continued until the ping time interval reaches the final interval rate, at which time all of the following pings will go out at the preset FinalPingInterval. The default value is 960 seconds.**WARNING: Please consult with iPass before changing any default ping interval values. Incorrect settings can significantly impact your network performance.**
	IdleTimeout: The connection's idle time before it is torn down. Valid range is 60000 to 300000 milliseconds. The default value is 300000 milliseconds (5 minutes).
	KeyStoreProperty: This is used for KeyStore configuration for ssl connection. The value is configured as KeyStore<n>, where n starts from 1 If KeyStoreProperty is not configured for a IpassServer, then NS will perform KeyStore failover logic based upon configuration sequence of KeyStore of KeyStoreType=ns. If KeyStoreProperty is configured for a IpassServer, then NS will first make use of the configured KeyStore and then try with the rest of the configured KeyStores of KeyStoreType=ns for KeyStore Fail over. All KeyStores will be iterated in Round Robin manner.
KeyStore	Provides KeyStore information. **Expected format**: KeyStore1 = name11=value11,name12=value12,...KeyStore2 = name21=value21,name22=value22,...
	KeyStoreType: This entry determines the java keystore type to genrate keystore for ns (SSL Connection) or eap (EAP-TLS).There can be max n and min 1 KeyStores of type ns There can be max 1 and min 0 KeyStores of type eap
	KeyStorePath=This entry determines the java keystore path. Default value for this property is set to /usr/ipass/netserver/current_version/certs/[KeyStoreName]
	KeyPassword : This entry determines the password required to get keys from java keystore Default value for this property is set to changeme
	KeyAlias : This entry determines the java keystore private key Alias Default value for this property is set to ns for KeyStoreType = NS or eapserver for KeyStoreType = EAP
	CertAlias : This entry determines the java keystore trusted certificate Alias Default value for this property is set to ipassca for KeyStoreType = NS or eapca for KeyStoreType = EAP
	KeyStorePassword: This entry determines the password required to open java keystore. Default value for this property is set to changeme
	Salt: This entry determines the salt used for encrypting KeyPassword and KeyStorePassword. Default value for this property is set to iPassNS
Listener	List of the Listeners for this server. **Expected format**: Listener1=Type=<protocol>,Port=<port number>,IpAddress=<local IP address> Listener2=Type=<protocol>,Port=<port number>,IpAddress=<local IP address>
	Default Listeners are: Listener1=Port=11812
	Listener2 = Port=11812,Type=Radius
	Listener3 = Port=11813,Type=Radius
	Listener4 = Port=11817,Type=RadiusProxy
	Listener5 = Port=11818,Type=RadiusProxy
	NumOfThreads: You can improve connectivity to a NetServer by increasing the number of threads accepting requests on port 11812. This can be helpful for if your NetServer in under heavier stress, such as 10 or more requests per second. For example: Listener1=Port=11812,NumOfThreads=10. This is an advanced setting. The server may not function properly if this value is set incorrectly.

LocalAccounting	LocalAccounting=<true>. This attribute, if set to true, enables the server to store the accounting START and STOP records locally. It normally stores in the detail.txt file under ipass.server.home/ipaddress of the machine. If it fails to create this file, it stores under ipass.server.home/logs.
LocalAccountingDir	LocalAccountingDir=<local accounting directory> A provider can enable local accounting (i.e. the detail.txt file for each RADIUS client or NAS with the LocalAccounting=true flag. This property allows them to customize the location of those detail.text files. Default value for this property is set to $ipass.server.home/s.
LogDirFileDeletionAge	LogDirFileDEletionAge=<age in days>. Valid range is: 0 to 180 days. The default is 90 days. A value of 0 means deletion is disabled. LogDirFileDeletionAge determines how old files in the directory <iPass Server Home>/logs must be before the are deleted. The check for file age is done only when the log file rotation happens.
MaxProxyTime	MaxProxyTime=<max proxy time in seconds>. This determines the maximum time for handling proxy requests. If a proxy reply is received that exceeds this limit then the RADIUS packet will be dropped. The property's value must be greater than 0 seconds and within 3600 seconds. Default value for this property is set to 30 seconds.
MultiProvider	MultiProvider= YES/NO. Default is set to NO. If enabled, the CustomerId sent to iPass will be that of the RadiusClient that the request came from. If the CustomerId is not set in the RadiusClient info, the main CustomerId of this server is used.
ProxyAcctServer	Provides RADIUS Proxy Server information.
	Sample format of the entires: ProxyAuthServer1=name11=value11,name12=value12,... ProxyAuthServer2=name21=value21,name22=value22,...
	ProxyAcctServer attributes:
	IpAddress: The RADIUS proxy server's hostname or IP address.
	Port: The proxy server's port number.
	SharedSecret: The shared secret used by the RADIUS proxy server.
	IncludeDomain: Include the user's domain in the request sent to the proxy server. The default is YES, always keep the domain with the username.
	ValidateAuthenticator: Specifies if the RADIUS Authenticator should be validated. Values are YES or No. Default is YES.
ProxyAuthServer	Provides RADIUS Proxy Server information.
	Sample format of the entries: ProxyAuthServer1=name11=value11,name12=value12,... ProxyServer2=name21=value21,name22=value22,...
	ProxyAuthServer attributes:
	IpAddress: The RADIUS proxy server's hostname or IP address.
	IncludeDomain: Include the user's domain in the request sent to the proxy server. The default is YES, always keep the domain with the username.
	ValidateAuthenticator: Specifies if the RADIUS Authenticator should be validated. Values are YES or NO. Default is YES.
RadiusClient1	Provides RADIUS client information. Only RADIUS clients listed here can send requests to this server.
	Sample format of the entries:
	RadiusClient1=name11=value11.name12=value12,... RadiusClient2=name21=value21,name22=value22,...
	RadiusClient attributes:
	IpAddress: The RADIUS client's IP address.
	SharedSecret: The shared secret used by the RADIUS Client.
	CustomerId: Used by multi-providers to specify an alternate iPass CustomerId.
	ValidateAuthenticator: Specifies if the RADIUS Authenticato should be validated. Values are YES or NO. Default is YES.

RoutingRealm	Specifies where to route the requests.Route either to the iPass Transaction Servers or proxy to a RADIUS server.Below are few examples
	RoutingRealm1=realm=IPASS,AuthServer=IpassServer,AcctServer=IpassServer,Strip=Yes
	RoutingRealm2=realm=IPASV,AuthServer=IpassServer,AcctServer=IpassServer,Strip=Yes
	RoutingRealm3=realm=DEFAULT,AuthServer=IpassServer,AcctServer=IpassServer
	RoutingRealm4=realm=NOREALM, AuthServer=ProxyAuthServer1, AcctServer=ProxyAcctServer1.Below is a list of various RoutingRealm attributes:
	Realm : The name of the realm to route based on.Use the keyword DEFAULT when specifying the default server(s) to use when the Realm(s) are not matched with the user's realm.Use the keyword NOREALM when specifying the server(s) to use when the username constains no realm/domain at the end of it.
	AuthServer: Specify the authentication server to use.When an IpassServer is used, it will use the entire IpassServerX list configured.
	AcctServer: Specify the accounting server(s) to use.When an IpassServer is used, it will use the entire IpassServerX list configured.
	Strip = Yes/No.Determines if the routing realm (Prefix or Suffix) need to be stripped or not from the NAI before sending the requests to IpassServers and ProxyServers.Default = No.
StartUpMessage	StartUpMessage=yes/no. This entry determines if a message is genterated by the server on startup. This is an advanced setting. The server may not function properly if this value is set incorrectly. Default value for this property is set to no(startup messages are turned off)
StatusTraceCollectInterval	StatusTraceCollectInterval=<number of minutes>. Minimum value: 60 minutes. Maximum value:1440 minutes. Default value: 60 minutes. StatusTraceCollectInterval determines the time interval between collection of statistics into the StatusTraceFile.
StatusTraceUploadInterval	StatusTraceUploadInterval=<upload frequency in minutes>. Minimum value: 120 minutes. Maximum value: 10080 minutes. Default value: 1440 minutes. StatusTraceUploadInterval determines the frequency of upload of status trace file.
StripRealm1	StripRealm1=<realm_name>. Where the <realm_name> is a domain name to be stripped away from the end of the username, such as: user@domain@extraDomain. This feature can be used to remove the extra domain some providers attached to the username.
TraceLogBackupType	TraceLogBackupType=<backupType>. Where <backupType> is either MultipleWithTimestamp or SinlgeBackup. The default is SingleBackup. TraceLogBackupType sets thr trace log's backup file name when rotation is to be performed on the local trace files.
TraceLogRotationHours	TraceLogRotationHours=<hours>. Valid range is: 1-720 hours. The default is 168 hours (1 week). TraceLogRotationHours controls how often the local trace file is rotated.
TraceLogRotationMaxSize	TraceLogRotationMaxSize=<max size>. Minimum value is 100 kB. Maximum value is 20000 kB. The default is 10000 kB. TraceLogRotationType sets the type of rotation to be performed on the local trace file(s).
UpdateInterval	UpdateInterval=<DayOfWeek Hour:Minute>. Where DayOfWeek ranges from Sunday to Saturday, and hour of the day is between 0-23. Default value is Monday 2:00. Determines when the Software Update module contacts the update server. The UpdateInterval mechanism resychronizes with the system clock every sixty minutes.
UpdateServer	Provides iPass software Update Server information.
	Sample format of the entries: UpdateServer1=name11=value11,name12=value12,... UpdaterServer2=name21=value=21,name22=value22,...
	UpdateServer attributes:
	IpAddress: The URL of the iPass software update server.
	RetryDelay: The time delay, in minutes, before retrying a server that recently failed a connection request. When a connection fails to a server, it is reordered to the end of the list. Once the RetryDelay expires, that server is brought back to the top of the list. The default value is 15 minutes. Valid range is: >=0.
	FailureThreshold: Once the failure count exceeds the Failurethreshold, the server is reordered to the end of the list. The default value id 0.
UploadAtStartup	UploadAtStartup=TRUE/FALSE. Default is set to TRUE. Determines if file upload should be done at startup. Note that this feature works in conjunction with AutoUpload. This feature will be disabled if AutoUpload is disabled.
UploadInterval	UploadInterval=<upload frequency in days>. Minimum value: 1 day. Maximum value: 7 days. Default value: 7 days. UploadInterval determines the frequency of upload of config, cert, status trace, and download trace files.

UploadServer	Provides iPass software Upload Server information.
	Sample format of the entries: UploadServer1=name11=value11, name12=value12,... UploadServer2=name21=value21,name22=value22,...
	UploadServer attributes:
	IpAdress: The URL of the iPass software update server.
	RetryDelay: The time delay, in minutes, before retry a server that recently failed a connection request. When a connection fails to a server, it is reordered to the end of the list. Once the RetryDelay expires, that server is brought back to the top of the list. The default value is 15 minutes. Valid range is: >=0.
	FailureThreshold: Once the failure count exceeds the FailureThreshold, the server is reordered to the end of the list. The default value is 0.
UseCalledStationIDForAuthCache	UseCalledStationIDForAuthCahce=y/n. This is an advanced setting. If this flag is enabled, Called-Station-ID will also be used for auth cache sensitivity.
UseEquantDnaForAuthCache	UseEquantDnaForAuthCache=y/n. This is an advanced setting. If this flag is enabled, equant_dna (first 4 bytes of Calling-Station-Id) will also be used for auth cache sensitivity.
UseIspCodeForAuthCache	UseIspCodeForAuthCache=y/n. This is an advanced setting. If this flag is enabled, the CostumerId (provider code) from the properties will also be used for auth cache sensitivity.
UseNasIpForAuthCache	UseNasIpForAuthCache=y/n. This is an advanced setting. If this flag is enabled, NAS-IP-Address will also be used for auth cahce sensitivity.
ZipLogFilesEnabled	ZipLogFilesEnabled=true/false. Determines whether or not trace and log files are compressed. Default is set to true.

Go to: **Other Product Documents** > **NetServer Admin Guide**

netserver

Running NetServer

This section describes a number of NetServer runtime commands. The following details are available:

Start NetServer

Typically the Netserver is installed in the iPass Product Directory /usr/ipass/netserver/ by version. The <NS_HOME> is linked to the running product version using a current_version symbolic link. e.g.

```
   NS_HOME = "/usr/ipass/netserver/current_version"
```

To start the NetServer manually:

1. Change directory to: <NS_Home>/bin
2. Type: ./netserverd start

Stop NetServer

To stop NetServer:

1. Change directory to: <NS_Home>/bin
2. Type: ./netserverd stop

Kill NetServer

You can also stop the NetServer by using the kill command:

- <NS_Home>/bin/netserverd kill. However, unlike the regular stop, this is a non-graceful stop and will immediately shut down any processes without closing them. It will also end all NetServer processes on the host, not just for the single NetServer. Only use the kill command if stop does not

work.

Restarting NetServer

To restart (stop and then start) NetServer:

1. Change directory to: <NS_Home>/bin
2. Type: ./netserverd restart

ns_command

You can also perform many runtime functions by using the tool ns_command, in the <NS_Home>/bin directory. ns_command can only be used locally, not remotely.

Usage: ns_command.csh <options>

Where your options are:

- **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 ipassNS.properties file. These are:
  - AutoUpdate flag, used to enable/disable automatic software update.
  - AAA Servers (AuthServer and AcctServer properties)
  - 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.
- **version**: Prints the server version.
- **file_upload**: Uploads the file named to the upload server.
- **force_log_rotation**: Causes the server to rotate/backup its log file.
- **sslc version**: Print the version of the SSL-C Library.

Note: For any KeyStore changes the NetServer restart is required to take effect.

Help

NetServer has a help tool, found in your <NS_Home>/bin directory, which you can use to get information on the configurable properties in the ipassNS.properties file.

To list all server properties, run: ipassconfig.csh -listall

To describe usage of a property, run: ipassconfig.csh -help <property name>

Log Files

There are several important log files associated with NetServer operations. **netserver.trace**, located in <NS_Home> contains daily traffic statistics, including:

- time
- number of authorization requests, accepts, challenges and rejections
- number of cache hits
- number of accounting starts, stops and updates.
- number of proxy requests
- The nsdownload.trace file, located in <NS_Home>/logs, records software download activities. It also contains the number of pending or corrupted accounting files on the local NetServer system.
- nsfailurecount: This log records any connection failures between NetServer and Transaction Servers and can help track which Transaction Servers have poor connectivity rates. (These messages will continue to also be logged in the netserver.trace file.) Connection failure messages only appear at DebugLevel=1 or greater. The TraceLogRotation properties will control when the file is backed up.

DebugLevel

The amount of debugging output in netserver.trace can be controlled by changing the value of the DebugLevel property. The range for this value is 0 to 5 (inclusive), where 0 produces the least amount of output, and 5 produces the highest.

Debug Level	Logging Output
0	Only severe problems logged.
1	Error messages.
2	Error and Debug messages.
3	Error, Debug, and Packet parsing information.
4	Error, Debug, Packet parsing, and Packet dumping.
5	Detailed Packet and Debug information.

- **Property**: DebugLevel
- **Default Value**: 0

iPass recommends a debug level of 3 in a production environment.

Log File Deletion

A DebugLevel of 5 produces a great deal of output. This can cause the NS.trace file to grow very large, and may slow the processing time of the NetServer. To control this, you can set log files to be deleted after a specified period of time.

- **Property**: LogDirFileDeletionAge
- **Default Value**: 180 <days>

Get Version Tool

You can check your NetServer version by running the Get Version tool.

To check your NetServer version, in <NS_Home>/bin, run ns_get_version.csh.

Automatic Software Updates

NetServer can be configured to periodically poll the iPass update server for the latest version of NetServer, and then automatically install it.

AutoUpdate

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

- **Default Value**: No
- **Valid Range**: Boolean

UpdateInterval

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

- **Default Value**: Monday 02:00
- **Valid Range**: <any day> <24 hour time>
- **To enable**: set AutoUpdate to Yes.

Transaction Center List Update

In addition to software updates, NetServer will periodically poll the iPass update server for the most current list of iPass transaction servers. The file is called TCList. If there is a change to the list, the new servers will automatically be added to the list in ipassNS.properties. This feature is enabled automatically and does not need to be set.

KeyStore Failover/Retrieval

KeyStoreProperty is an optional attribute for IpassServer. If only one NS keystore has been provisioned, the attribute is not required. If there are multiple ns-type keystores available, this attribute should be included to indicate the “initial” keystore to use.

If **KeyStoreProperty** is not configured for an IpassServer with multiple keystores, then the NetServer will perform KeyStore failover logic based on the configuration number sequence order of KeyStoreType=“ns” in order to match it with one acceptable to the iPass Transaction Server. For example in ipassNS.properties Ipassservers configured and Keystores list as:

```
IpassServer1=IpAddress=<server ip>,Port=<server host>
IpassServer2=IpAddress=<server ip>,Port=<server host>

KeyStore1=KeyStoreType=eap,KeyStorePath=$ipass.server.home/certs/eapserver.k
eystore,KeyPassword=UfGjld0YWEUjEIZUnNvIsA==,KeyStorePassword=UfGjld0YWEUjEI
ZUnNvIsA==
KeyStore2=KeyStoreType=ns,KeyStorePath=/usr/ipass/netserver/current_version/
certs/ns1.keystore,KeyAlias=ns,CertAlias=ipassca,Salt=iPassNS,KeyPassword=XP
avlIhNARgjEIZUnNvIsA==,KeyStorePassword=XPavlIhNARgjEIZUnNvIsA==
KeyStore3=KeyStoreType=ns,KeyStorePath=/usr/ipass/netserver/current_version/
certs/ns3.keystore,KeyAlias=ns,CertAlias=ipassca,Salt=iPassNS,KeyPassword=Vu
6viZxGH30jEIZUnNvIsA==,KeyStorePassword=Vu6viZxGH30jEIZUnNvIsA==

IpassServer1 tries to make connection with TServer using KeyStore2 ('ns'
type keystore appearing 1st in the list).If fails to make connection due to
SSL handshake error with KeyStore2 it then routes to KeyStore3. All
KeyStores will be iterated in Round Robin manner.If in ipassNS.properties
Ipasservers are configured as :
IpassServer1=IpAddress=<server ip>,Port=<server host>,KeyStore1=KeyStore3
```

In this case, IpassServer1 tries to make connection with TServer using KeyStore3 (as it is configured with IpassServer1).If fails to make connection due to SSL handshake error with KeyStore3 it then routes to KeyStore2 then again to KeyStore3.

Go to: **Other Product Documents > NetServer Admin Guide**

netserver

Sample ipassNS.properties File

```
# File: ipassNS.properties.example

# Description: iPass NetServer configuration file.

# Blank lines and lines beginning with # ignored.



# This file contains a subset of the most commonly used properties.

# For a complete listing of all available properties,
# go to the directory <ns-home>/bin/ and execute the
# command: "ipassconfig.csh -listall"

# For a detailed description of a particular property,
# go to the directory <ns-home>/bin/ and execute the
# command: "ipassconfig.csh -help <property name>"


# Your iPass Customer ID

CustomerId=1



# Configure RadiusClients

###RadiusClient1=ipaddress=10.10.6.2,sharedsecret=testkey
###RadiusClient2=ipaddress=10.10.50.19,sharedsecret=testkey



# Configure MultiProvider
# Determines if MultiProvider functionality is enabled.
# If enabled, the CustomerId sent to iPass will be that of the RadiusClient
# that the request came from.
# If the CustomerId is not set in the RadiusClient info, the main
# CustomerId of this server is used.
# Eg: to set a customerId for a client using RadiusClient settings:
# RadiusClient1=ipaddress=10.10.6.2,sharedsecret=testkey,CustomerId=111

###MultiProvider=Yes


# Mapping Realm to ProxyServer(s)
```

```
RoutingRealm1=realm=IPASS,AuthServer=IpassServer,AcctServer=IpassServer,Strip=Yes
RoutingRealm2=realm=IPASV,AuthServer=IpassServer,AcctServer=IpassServer,Strip=Yes
##RoutingRealm3=realm=DEFAULT,AuthServer=IpassServer,AcctServer=IpassServer
##RoutingRealm4=realm=NOREALM,AuthServer=ProxyAuthServer1,AcctServer=ProxyActServer1



# Proxy Server settings
# Protocol should be defaulted to Radius

###ProxyAuthServer1=protocol=RADIUSProxy,ipaddress=127.0.0.1,port=1812,IdleTimeout=15000,sharedsecret=testkey
###ProxyAcctServer1=protocol=RADIUSProxy,ipaddress=127.0.0.1,port=1813,IdleTimeout=15000,sharedsecret=testkey



# Ipass Server (Transaction Server List)

IpassServer1=IpAddress=auth7.ipass.com,Port=9101,KeyStoreProperty=KeyStore2
IpassServer2=IpAddress=auth8.ipass.com,Port=9101,KeyStoreProperty=KeyStore2
IpassServer3=IpAddress=auth-apac.ipass.com,Port=9101,KeyStoreProperty=KeyStore2
IpassServer4=IpAddress=auth-sjc.ipass.com,Port=9101,KeyStoreProperty=KeyStore2




# Auth, Acct, and Proxy Listener information.

# Sample line:
# Listener1= Port=<value>
# Port - Port number to listen for iPass requests from.
# Default is UDP port 11812/11813.



Listener1=Type=Radius,Port=11812
Listener2=Type=RadiusProxy,Port=11817
Listener3=Type=SSLPost,Port=11811



# IP Addresses, in X.X.X.X format, permitted to send control messages (such as
#    shutdown and restart) to this server. Multiple IPs can be specified. All
```

```
#    must be unique and contain the prefix ControlMessageIp.
#    By default, the local host and iPass Transaction Servers IP address
#    are already included.

# Sample format:
# ControlMessageIp1=555.555.555.555
#



# Debug level determines if debug and error messages are logged
# to the event table.
#   Debug Level 0 - Only severe messages are logged
#   Debug Level 1 - Error messages are logged
#   Debug Level 2 - Error and Debug messages are logged
#   Debug Level 3 - Error, Debug, and Packet parsing information is logged
#   Debug Level 4 - Error, Debug, Packet parsing, and Packet dumping is
logged
#   Debug Level 5 - Detailed Packet and debug information is logged

# Note: Production servers should normally run with debug level 0 or 1.

DebugLevel=0

AutoUpload=yes
UploadAtStartup=yes
AutoUpdate=no

# Allow Accounting Update Messages to Pass-through to TS
AllowAcctUpdate=yes




# EapMode determines if the NetServer will do early-termination
# of EAP-TTLS requests. Primarily EAP-TTLS/PAP.
# All other EAP types will be blocked unless otherwise configured to do so.
# Default setting is: yes or true.

EapMode=YES



# EapNotification determines if the NetServer will send back
# the Reply-Message(s) in EAP-Notification Requests prior to
# sending back the final Radius Access-Accept/Access-Reject.
# Default setting is: true

EapNotification=NO



# This feature is used in conjuction with the EapNotification feature.
```

```
# It is used to filter which Reply-Message(s) can get sent back
# as EAP-Notifications. It will check if any Reply-Messages begin
# with the given FilterPrefix string.
# FilterPrefix: The string to match at the beginning of the Reply-Message.
#               It is case insensitive.
# KeepPrefix: Whether to keep that prefix attached to the Reply-Message
#             when sending back as an EAP-Notification.


EapNotificationFilter1= FilterPrefix="Location=",KeepPrefix=YES

# Tunneled EAP NAI Check
# This function ensures the declared EAP Identity of an EAP-TTLS
# request used in accounting without CUI, eventually contains within
# the username@domain NAI of the Authorized Account negotiated with iPass
# during secondary authentication phase.  Prevents Tunnel Fraud.

EapNaiCheck=true



# EAP Early Termination
# With EapMode enabled, the netserver offers native EAP-TTLS tunneling to
# the iPass AAA fabric.  The tunnel is protected by 2048-bit RSA encryption
# secured by Thawte global.  This provides a single trust point,
EAP-Identity
# Token and Authentication Method for a connection profile on any iPass
enabled network.
# The "inside" secondary auth method is performed with the Home AAA via the
# TCP/SSL iPass Transaction Network (PAP, MSHCAPv2, GTC, TLS, EAP-SIM) by
# support or preference behind the Roamserver. Suitable for fast roaming.



EapEarlyTerminate=21

# EAP Pass-Through Filters
# If EapMode is yes, TTLS will never pass-through the Netserver.  For
default
# pass-through, set EapMode=no. With EapMode Enabled, EapPassThroughAllow
# specifies the EAP Protocol Id which are allowed to transact end-to-end
# via the iPass Transaction Network.  Additional Network Policies Apply.
# End-to-end EAP Authentication methods can be prohibitively slow.


# EapPassThroughAllow=4,6,13,23,25,43
# EapPassThroughAllow=all
# EapPassThroughDeny=nothing



# CUI SETTINGS
```

```
#
# If EapNaiCheck is false, a supplier must support attribute 89
# in all Radius Accounting.  These settings enable reflection of
# Chargeable-User-Identity Attribute 89 in all Access-Accept.
# The value of the attribute is the accepted userid negotiated with iPass
# Transaction Centers unless a CUI is returned by the Home AAA.


CuiEnable=yes

#
# CuiHasEnable encodes the value of the CUI returned in an Access-Accept to
# create a unique and anonymous identity hash of the user portion of the NAI.
# This hash is reversable by iPass for billing correlation.

CuiHashEnable=no


#
# CuiHashingStrategy sets the type of hashing strategy to be performed
# on CUI hashing. That is, either the complete NAI or just the USER_NAME
# in an NAI to be hashed.


CuiHashingStrategy=USER_NAME

# CuiAcctUserReplace retrieves the CUI from all Accounting messages and
# replaces the Anonymous EAP Identity token used for routing the transaction
in
# the User-Name with the value of the CUI returned in the Access-Accept.
# If CuiHashEnable is yes, CuiAcctUserReplace can optionally leave the
User-Name
# encoded for transmission to iPass for correlation by the Home AAA without
CUI.


# CuiAcctUserReplace1=Token=all,Decode=yes
CuiAcctUserReplace1=Token="IPASS/user@ipass.com",Decode=yes
CuiAcctUserReplace2=Token="user@ipass.com",Decode=Yes
CuiAcctUserReplace3=Token="IPASS/user@ipass",Decode=yes
CuiAcctUserReplace4=Token="user@ipass",Decode=Yes
CuiAcctUserReplace5=Token="IPASS/anonymous@ipass.com",Decode=yes
CuiAcctUserReplace6=Token="anonymous@ipass.com",Decode=yes
CuiAcctUserReplace7=Token="IPASS/anonymous@ipass",Decode=yes
CuiAcctUserReplace8=Token="anonymous@ipass,Decode=yes


# CACHE DEFAULTS

# Determines if the caching of successful authentication requests is enabled
AuthCacheEnabled=True
```

```
# Auth Cache. Limit by the number of users.
AuthCacheSize=60

# Auth Cache days Limit by the number of days in cache
AuthCacheDays=1


#Filesize rotation
LocalAccounting=false
AcctLogBackupType=MultipleWithTimestamp
AcctLogRotationMaxSize=10240
AcctLogRotationType=FileSize


TraceLogBackupType=MultipleWithTimestamp
#TraceLogRotationType=NumberOfHours
#TraceLogRotationHours=24
TraceLogRotationType=FileSize
TraceLogRotationMaxSize=10240
LogDirFileDeletionAge=120


# Determines the Keystore which will be used for administrative purpose.
# The configured KeyStore must be of KeyStoreType=ns

AdminKeyStoreProperty=KeyStore2


KeyStore1=KeyStoreType=eap,KeyStorePath=$ipass.server.home/certs/eapserver.k
eystore,
KeyPassword=UfGjld0YWEUjEIZUnNvIsA==,KeyStorePassword=UfGjld0YWEUjEIZUnNvIsA
=

KeyStore2=KeyStoreType=ns,KeyStorePath=$ipass.server.home/certs/ns1.keystore
```

Go to: **Other Product Documents** > **NetServer Admin Guide**

netserver

Third-Party RADIUS Configurations

This section provides configuration instructions for several different third-party RADIUS products. These configurations will allow the RADIUS server to route iPass traffic to the NetServer, which will route to the iPass Transaction Centers for authentication. Use these instructions only when configuring RADIUS in environments where the NetServer is installed behind the RADIUS server.

If your network configuration requires the NetServer to be in any other location relative to your NAS and RADIUS servers, you will need to change your configuration accordingly. For further information on this, please consult the documentation provided with your server software.

This help page includes information on the following:

NetServer supports many varieties of RADIUS server. Instructions found here do not imply that iPass endorses a particular RADIUS solution. We only provide information on these types as a helpful reference as it relates to NetServer operation. Always consult your RADIUS server's documentation for the most current and complete information on configuring your RADIUS server.

RADIATOR

iPass providers who use RADIATOR can choose between two different methods of configuration.

Configuring RADIATOR Using the IPASS/ Prefix

To configure RADIATOR to route iPass traffic based on the IPASS/ prefix, you will need to alter your RADIATOR configuration file, radius.cfg.

1. Add entries to the clients list in the radius.cfg file.

In the radius.cfg file (/etc/raddb/radius.cfg), there will be a section containing your clients list. For each client, this file will have a section that looks similar to the example below. To allow RADIATOR to route iPass traffic to the NetServer, add the new italicized line here to the very bottom of every distinct client entry in this file:

```
<Client 123.456.789.0>
```

```
Secret the-secret-we-share-with-NAS's


RewriteUsername
s/^IPASS%%\%%///([^@]+)//%%\%%@([^@]+)$/IPASS<nowiki>\</nowiki>/
class="code"# <Client 123.456.789.0> Secret the-secret-we-share-with-NAS's
RewriteUsername
s/^IPASS%%\%%///([^@]+)//%%\%%@([^@]+)$/IPASS<nowiki>\</nowiki>/$1#$2<nowiki
>\</nowiki>@myipass/ </Client> <nowiki>\</nowiki>@myipass/


</Client>
```

This entry will allow RADIATOR to append @myipass to the username of all iPass users. In addition, the first @ in the username will be changed to a # sign.

2. Add entries to the Realm list in the radius.cfg file.

In the radius.cfg file (/etc/raddb/radius.cfg), there will also be a section containing your realm list. This section lists all of the realms known to RADIATOR, and defines how they are handled. Add the following entry to the realm list section. It can be placed anywhere within the section, provided it is placed above the DEFAULT realm entry.

```


<Realm myipass>


AcctLogFileName %L/ipass/detail


RewriteUsername
s/^IPASS%%\%%///([^#]+)//%%\%%//#([^@]+)//%%\%%@myipass$/IPASS<nowiki>\</now
iki>/ class="code"<nowiki>\</nowiki>@ <Realm myipass> AcctLogFileName
%L/ipass/detail RewriteUsername
s/^IPASS%%\%%///([^#]+)//%%\%%//#([^@]+)//%%\%%@myipass$/IPASS<nowiki>\</now
iki>/$1<nowiki>\</nowiki>@$2/ <AuthBy RADIUS> Host 123.456.789.0 AuthPort
11812 AcctPort 11813 Secret mysecret </AuthBy> </Realm myipass> /


<AuthBy RADIUS> Host 123.456.789.0


AuthPort 11812


AcctPort 11813


Secret mysecret
```

```
</AuthBy>

</Realm myipass>
```

This entry instructs RADIATOR to handle the @myipass realm by stripping the @myipass off the username and rewriting it in its original format. This means that we do not need the default realm and our proxy will be handled before any handler clauses.

The shared secret listed in the entry above must be the same value as the secret of the NetServer found in the ipassNS.properties file of your NetServer.

When you have finished editing radius.cfg, save and exit the file. Then restart RADIATOR to allow these changes to take effect.

Configuring RADIATOR Using the DEFAULT Realm

If it is not possible to configure RADIATOR to recognize the IPASS/ prefix (for example, if you are using an older version of the software), you may opt to route iPass traffic based on a DEFAULT realm. You may only use this option if you are not already using the DEFAULT realm, and you have defined all other realms for which traffic is received by RADIATOR.

If not all other realms are defined, all users with undefined domains will be routed to the NetServer. To use this configuration, add the following entry to as the final realm in the Realm section of the radius.cfg file (/etc/raddb/radius.cfg):

```
<Realm DEFAULT>

<AuthBy RADIUS>

Host 123.456.789.0

AuthPort 11812

AcctPort 11813

Secret mysecret

</AuthBy>
```

```
</Realm>
```

The shared secret listed in the entry must be the same value as the secret of the NetServer found in the ipassNS.properties file of your NetServer.

When you have finished, restart RADIATOR to allow these changes to take effect.

FreeRADIUS

iPass providers using FreeRADIUS will need to edit <path to radius>/raddb/sites-enabled/default, <path to radius>/raddb/modules/realm, and the <path to radius>/raddb/proxy.conf configuration files to allow iPass traffic to travel through their network.

1. Edit the realm section of your **<path to radius>/raddb/modules/realm** file.

Within the <path to radius>/raddb/sites-enabled/default, there will be a section containing your realm list. This section lists all of the realms known to FreeRADIUS, and defines how they are handled. To enable FreeRADIUS to recognize the IPASS/ prefix, make sure the following is uncommented in the realm file (and please add it if it is not present):

```
realm IPASS {

format = prefix

delimiter = "/"

}
```

2. Edit the authorization section of your **<path to radius>/raddb/sites-enabled/defaul** file.

Within the <path to radius>/raddb/sites-enabled/default file, there will also be an authorization section. This section defines how FreeRADIUS will authorize users. You will want to ensure that the listings in this section are in the order shown below to allow FreeRADIUS to perform authorization properly. The entry below allows FreeRADIUS to preprocess all users against the hints or huntgroups files, then to process all realms, and finally to look in the users file. The order of the realm modules will determine the order in which the FreeRADIUS will try to find a matching realm. You will need to add an entry for the IPASS/ prefix above the line for the suffix to allow these users to be processed first. When complete, this section should look similar to the example below:

```
authorize {
```

```
preprocess

IPASS

suffix

files

}
```

3. Edit the pre-accounting section of your **<path to radius>/raddb/sites-enabled/default** file. Another section you will need to edit in the <path to radius>/raddb/sites-enabled/default file is the pre-accounting section. The following entry allows FreeRADIUS to look for a proxy realm in the order that each realm is listed, then to look at the acct_users file, and finally to preprocess users using the hints file. You will need to add an entry for the IPASS/ prefix above the line for the suffix to allow these users to be processed first. When complete, this section should look similar to the example below:

```
preacct {

IPASS

suffix

files

preprocess

}
```

When you have finished editing radiusd.conf, save and exit the file.

4. Edit the users file.

The users file (/etc/raddb/users) dictates how FreeRADIUS authenticates users. You will need to ensure that there is a DEFAULT entry in the users file similar to the one shown below. Please note that this is only an example of the type of entry needed. If you already have a default entry, please let your iPass technician know what it is before modification:

```
DEFAULT Auth-Type = Local
```

When you have finished editing the users file, save and exit the file.

5. Add the IPASS/ realm entry to your proxy.conf file.

To complete this configuration and allow FreeRADIUS to proxy iPass traffic to your NetServer, you must add an entry for the IPASS/ prefix realm to your proxy.conf file(/etc/raddb/proxy.conf). The following entry can be to this file anywhere within the list of realm entries, provided it is placed above the DEFAULT realm entry.

```
realm IPASS {

type = RADIUS

authhost = IP.Address.of.NetServer:11812

accthost = IP.Address.of.NetServer:11813

secret = mysecret

nostrip

}
```

The shared secret listed in the entry must be the same value as the secret of the NetServer found in the ipassNS.properties file of your NetServer.

When you have finished editing proxy.conf, save and exit the file.

6. When complete, restart your FreeRADIUS to allow these changes to take effect.

DTC RADIUS

iPass providers using the DTC RADIUS software will need to add the an entry to their users (/etc/raddb/users) file to allow iPass traffic to travel through their network. In addition, the DTC RADIUS and the NetServer must be installed on different hosts, and they must use the same port number for routing requests (that is, if the DTC is sending requests on port 1812, the NetServer must run on 1812 on another host).

1. To allow the DTC RADIUS to recognize iPass users based on the IPASS/ prefix, and proxy these requests to the NetServer, add the following entry to your users file (/etc/raddb/users):

```
DEFAULT Password = "PROXY", Prefix = "IPASS/", DTC-Trunc-PreSuffix =
Trunc-No,

DTC-Limit-Login = Limit-No

DTC-Auth-Server = IP.Address.of.NetServer,

DTC-Acct-Server = IP.Address.of.NetServer,

DTC-Auth-Secret = "sharedsecret",

DTC-Acct-Port = 1813

DTC-Acct-Secret = "sharedsecret"
```

The shared secret listed must be the same value as the secret of the NetServer found in the ipassNS.properties file of your NetServer.

When you have finished editing the users file, save and exit the file.

2. When complete, restart your DTC RADIUS to allow these changes to take effect.

Cistron RADIUS

iPass providers using Cistron RADIUS will need to edit the clients, realms, and users configuration files to allow iPass traffic to travel through their network.

1. Edit the clients file.

The clients file (/etc/raddb/clients) contains a separate entry for each software application that acts as a client of Cistron RADIUS. To add the NetServer as a client of your RADIUS, add this entry to this file:

```
<IP.Address.of.NetServer> <SharedSecret>
```

The shared secret must be the same value as the secret of the NetServer found in the ipassNS.properties file of your NetServer.

When you have finished editing, save and exit the file.

2. Edit the realms file.

The realms file (/etc/raddb/realms) lists all of the realms known to Cistron RADIUS, and defines how they are handled. To enable the Cistron RADIUS to route iPass traffic using the DEFAULT realm, add these two lines to anywhere in this file.

```
NULL LOCAL

DEFAULT <IP.Address.of.NetServer>:11812 NOSTRIP
```

When you have finished editing, save and exit the file.

3. Edit the users file.

The users file (/etc/raddb/users) dictates how Cistron RADIUS authenticates users. You will need to ensure that there is a DEFAULT entry in the users file similar to the one shown below. Please note that this is only an example of the type of entry needed. If you already have a default entry, please let your iPass technician know what it is before modification:

```
DEFAULT Auth-Type = Local
```

When you have finished editing, save and exit the file.

4. Restart your Cistron RADIUS to allow these changes to take effect.

Go to: **Other Product Documents** > **NetServer Admin Guide**

netserver