Avaya IP Telephone File Server Application Reference Guide

16-601433
Issue 2
November 2006

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document may be incorporated in future releases.

For full legal page information, please see the complete document, Avaya Legal Page for Hardware Documentation, Document number 03-600759.

To locate this document on our Web site, simply go to http://www.avaya.com/support and search for the document number in the search box.

Documentation disclaimer

Avaya Inc. is not responsible for any modifications, additions, or deletions to the original published version of this documentation unless such modifications, additions, or deletions were performed by Avaya. Customer and/or End User agree to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation to the extent made by the Customer or End User.

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites referenced elsewhere within this documentation, and Avaya does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all of the time and we have no control over the availability of the linked pages.

Copyright

Except where expressly stated otherwise, the Product is protected by copyright and other laws respecting proprietary rights. Unauthorized reproduction, transfer, and or use can be a criminal, as well as a civil, offense under the applicable law.

Avaya support

Avaya provides a telephone number for you to use to report problems or to ask questions about your product. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com/support

Software License

USE OR INSTALLATION OF THE PRODUCT INDICATES THE END USER'S ACCEPTANCE OF THE TERMS SET FORTH HEREIN AND THE GENERAL LICENSE TERMS AVAILABLE ON THE AVAYA WEB SITE AT http://support.avaya.com/LicenseInfo/ ("GENERAL LICENSE TERMS"). IF YOU DO NOT WISH TO BE BOUND BY THESE TERMS, YOU MUST RETURN THE PRODUCT(S) TO THE POINT OF PURCHASE WITHIN TEN (10) DAYS OF DELIVERY FOR A REFUND OR CREDIT.

Avaya grants End User a license within the scope of the license types described below. The applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the Documentation or other materials available to End User. "Designated Processor" means a single stand-alone computing device. "Server" means a Designated Processor that hosts a software application to be accessed by multiple users. "Software" means the computer programs in object code, originally licensed by Avaya and ultimately utilized by End User, whether as stand-alone Products or pre-installed on Hardware. "Hardware" means the standard hardware Products, originally sold by Avaya and ultimately utilized by End User.

License Type(s):

Designated System(s) License (DS). End User may install and use each copy of the Software on only one Designated Processor, unless a different number of Designated Processors is indicated in the Documentation or other materials available to End User. Avaya may require the Designated Processor(s) to be identified by type, serial number, feature key, location or other specific designation, or to be provided by End User to Avaya through electronic means established by Avaya specifically for this purpose.

Third-party Components

Certain software programs or portions thereof included in the Product may contain software distributed under third party agreements ("Third Party Components"), which may contain terms that expand or limit rights to use certain portions of the Product ("Third Party Terms"). Information identifying Third Party Components and the Third Party Terms that apply to them is available on Avaya's Web site at:

http://support.avaya.com/ThirdPartyLicense/

Interference

Using a cell, mobile, or GSM telephone, or a two-way radio in close proximity to an Avaya IP Telephone might cause interference.

Licensing

The IP Telephone File Server Application is provided under license terms defined in the distribution package. Refer to the file Avaya_License.txt in the installation directory.

You will be prompted to accept the Avaya license terms during the installation and/or download. By accepting the license terms during the installation, you agree to be bound by its terms. If you do not wish to be bound by the license terms, you should decline the license terms and the installation will abort.

Warranty

In principle the software is provided on a free to use, but without any implied warranty basis and is unsupported unless explicitly agreed otherwise by Avaya in writing. Contact Avaya Global Services for information on the range of product support offers.

Trademarks

All third party trademarks are acknowledged.

Contents

Chapter 1: Introduction

The Avaya IP Telephone File Server Application provides IP telephone support as well as administrative server support.

For the IP telephones, the application provides the following:

- An HTTP server to support:
  - configuration file and firmware downloads to Avaya one-X™ Deskphone Edition 9600 Series IP Telephones,
  - configuration file and firmware downloads to Avaya 4600 Series IP Telephones, and
  - backup/restore of user-specific data for Avaya one-X™ Deskphone Edition 9600 Series IP Telephones.
- An HTTPS server to support:
  - configuration file downloads to Avaya one-X™ Deskphone Edition 9600 Series IP Telephones, and
  - configuration file downloads to Avaya 4600 Series IP Telephones.
- A TFTP server to support:
  - configuration file and firmware downloads to Avaya 4600 Series IP Telephones.
- An FTP server to support:
  - backup/restore of user-specific data for Avaya 4600 Series IP Telephones.

To support administrative functions, the application provides the following:

- Linux or Windows operating system versions run as Daemons/Services.
- Web-based status and administration capability.
- Detailed logging.
- TFTP “session” control for 4600 Series IP Telephones, to improve download reliability.
- SNMP lookup for anonymous, yet secure FTP login with zero administration, for 4600 Series IP Telephones only.
- IP address list control for selective downloading.
- Secure Web management option using HTTPS.
- Secure backup/update file server mode using TLS.
- Multi threading for server level performance.
- Optional “scanning” of installed IP Telephone firmware versions.

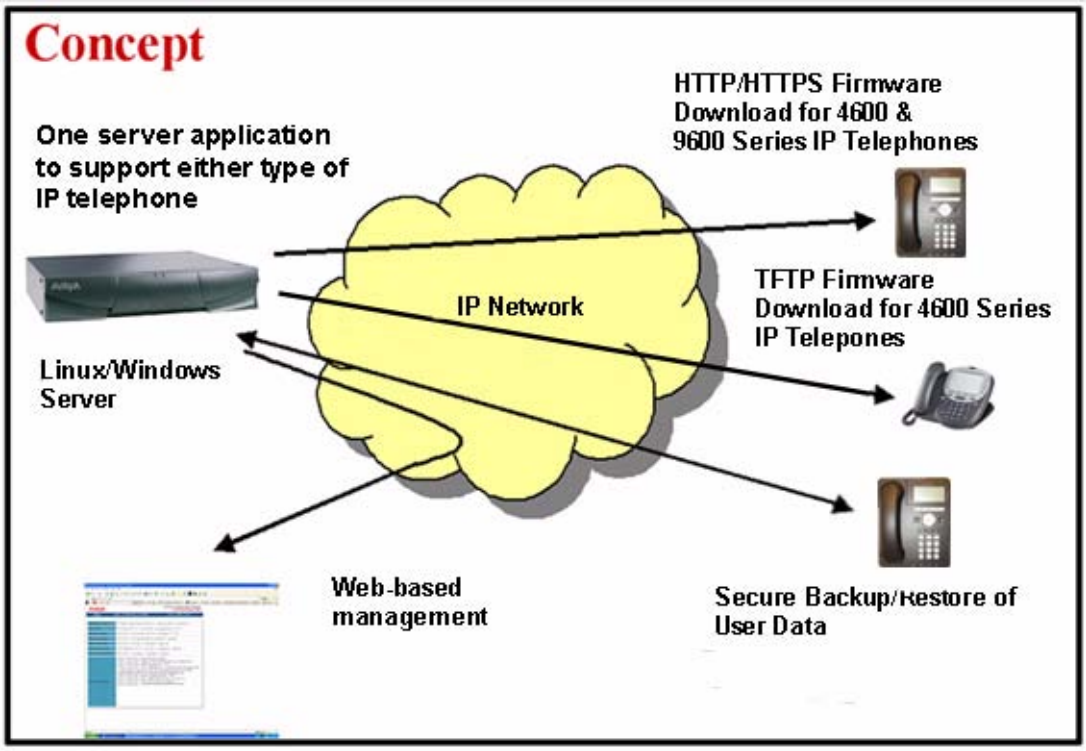
Combine the Avaya IP Telephone File Server Application with the DHCP functions of a RedHat Linux or Windows server for a single server solution for 4600 and 9600 Series IP Telephone system setup requirements.

Important:

This document covers the 4600 Series IP Telephones and the 9600 Series IP Telephones. References to TFTP and FTP apply **only** to 4600 Series IP Telephones. References to HTTP and HTTPS apply to **both** series of telephones, except for HTTP backup/restore, which applies **only** to 9600 Series IP Telephones.

HTTPS capabilities can only be used in conjunction with a valid Avaya certificate. Avaya HTTPS certificates are supplied as part of the MultiVantage™ Express distribution only. Therefore, if you deploy the Avaya IP Telephone File Server Application as a standalone solution, HTTPS capabilities are **not** currently supported.

Figure 1: Operational Concept



There are specific application extensions to:

- Define which IP addresses can download software for initial, limited deployment testing.
- Proactively monitor the server with a dedicated WatchDog process.
- Provide a highly available server pair under Linux using HeartBeat.
- Unpack and distribute IP telephone firmware automatically using secure HTTPS protocol.
- Use secure HTTPS protocol to back up user data to central file servers.

The core File Server Application works with two optional supporting applications:

- MV_Manager, which provides Web-based administration, and
- MV_WatchDog, which monitors the health of the server application.

Chapter 2: Modes of Operation

Introduction

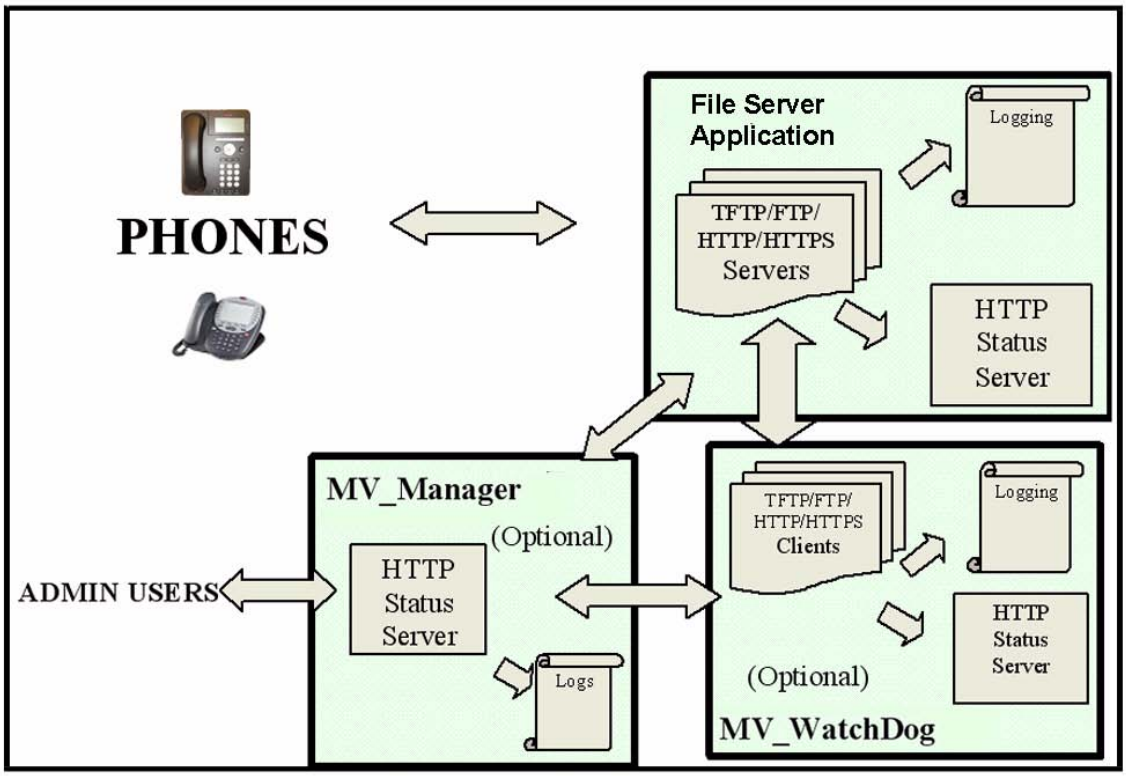
You can configure the IP Telephone File Server Application to operate in any of five ways:

- As a basic HTTP/HTTPS server for 9600 Series IP Telephones,
- As a basic TFTP/FTP/HTTP/HTTPS server for 4600 Series IP Telephones,
- As a basic server plus a file server client to send FTP backup and telephone firmware update requests to/from a central file server over TLS,
- As the central file server for the Enterprise,
- As both a basic server and central file server at the same time.

The following diagrams depict the architecture and typical application scenarios.

Basic Operation

Figure 2: Mode 1 - Basic Operation Mode



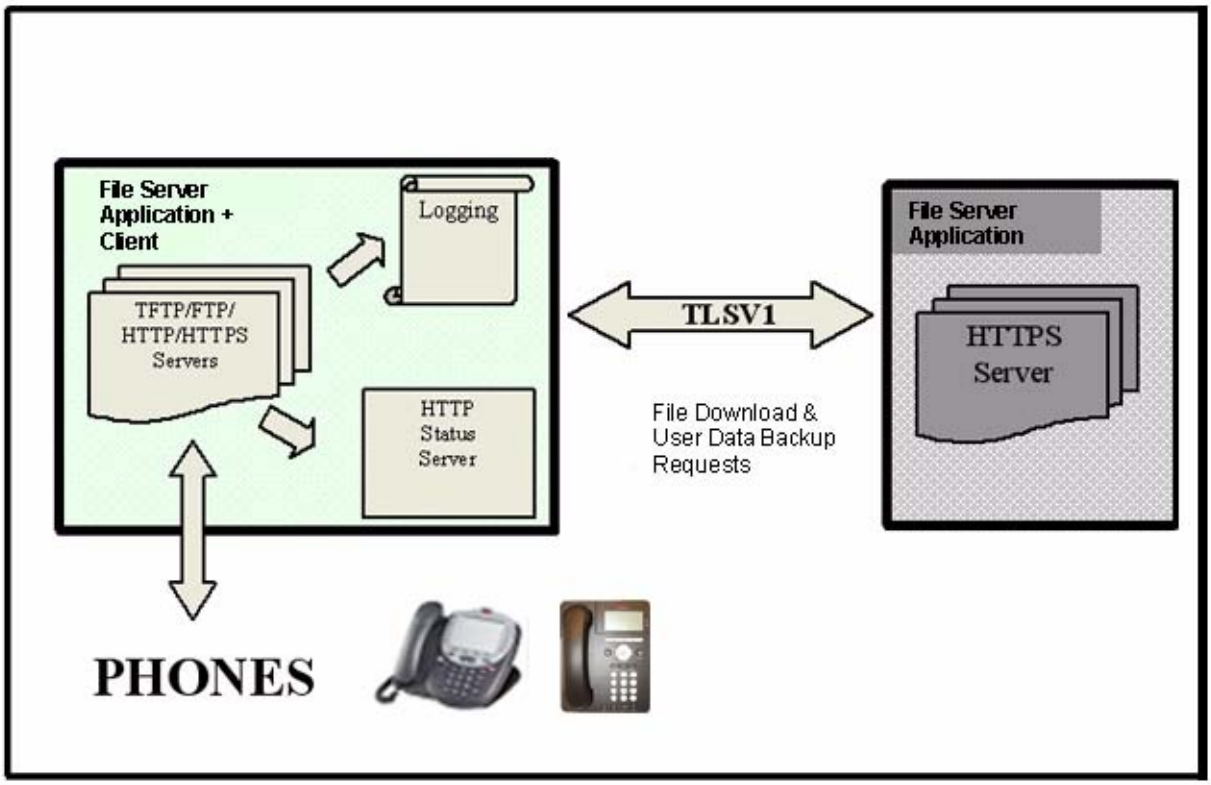
Use Case Scenario

Use the basic operation mode to:

- Download configuration files like the 96xxupgrade.txt and 46xxsettings.txt files to the telephones using HTTPS,
- Download software code files, for example, .bin, to the telephones using HTTP,
- Manage the server files locally, and
- Manage and monitor the server remotely using a standard Web browser.

Basic Server + File Server Client

Figure 3: Mode 2 - Basic Server plus File Server Client Mode



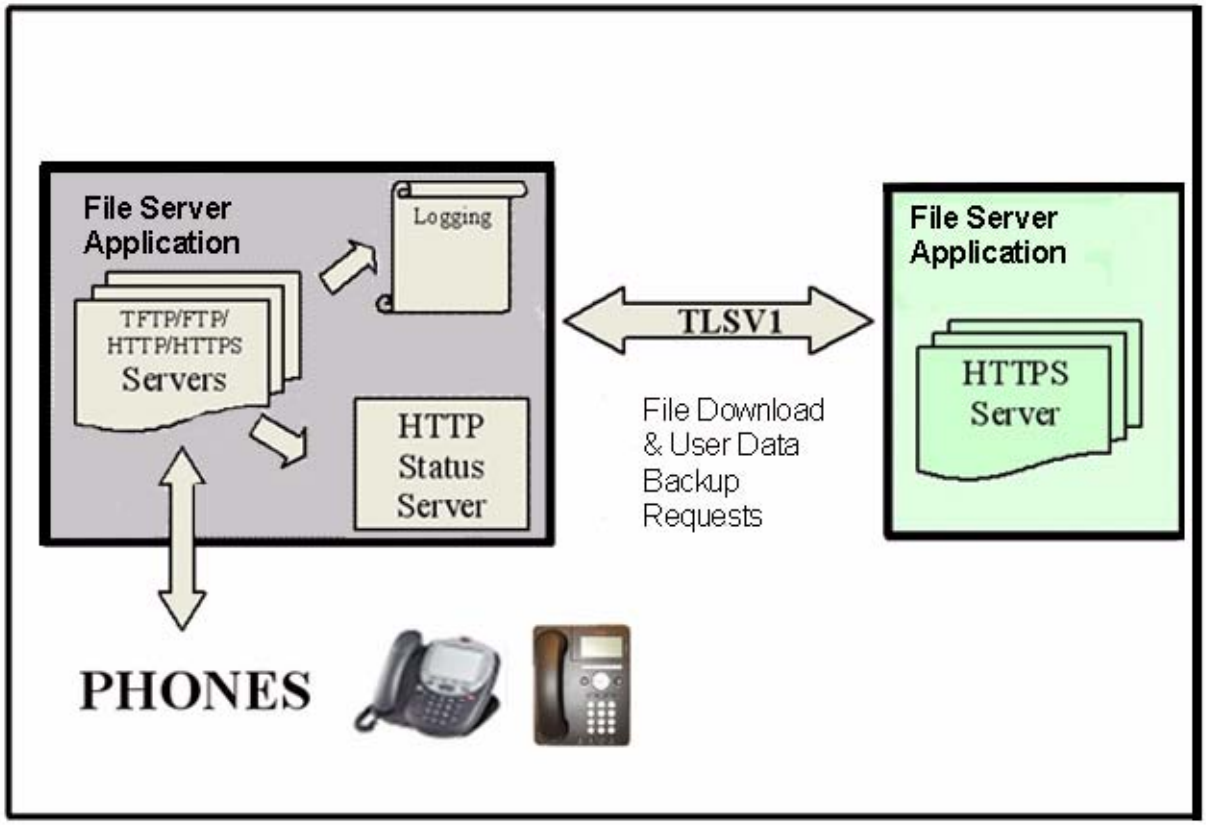
Use Case Scenario

Use the basic server plus file server client to:

- Function as a central Linux server delivering/backing up files,
- Provide a mix of Linux and Windows servers,
- Periodically request firmware updates from primary/secondary file servers,
- Automatically unzip new software to a ready for distribution state,
- Periodically store user data backup files on the central file server, and
- Trigger central user data backup and restore requests from the file server for backup (4600 Series IP Telephones only) and network hot desking.

Central File Server

Figure 4: Mode 3 - Central File Server Mode



Use Case Scenario

Use the central file server for the client mode to:

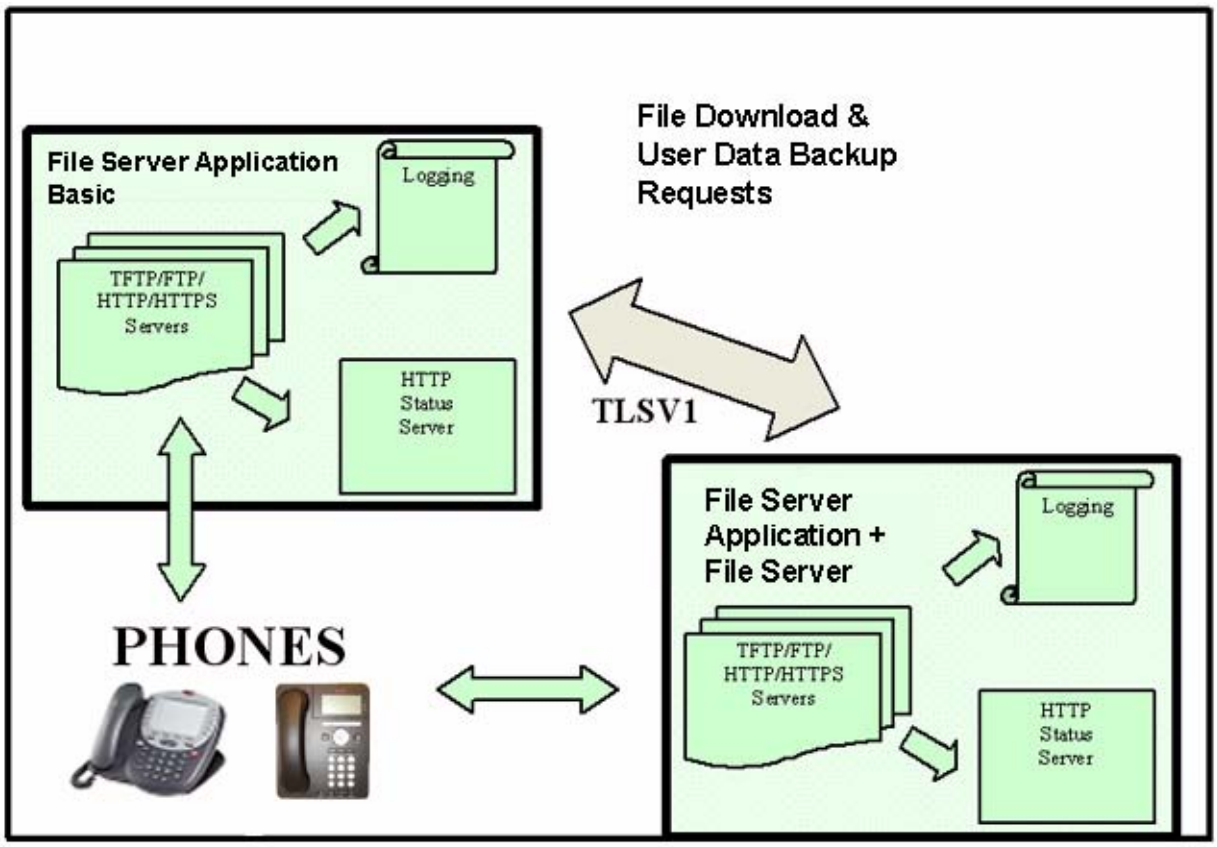
- Act as the central server, delivering update files and storing backing up files,
- Provide a mix of Linux and Windows servers,
- Use TLS for distribution security,
- Optionally request Client Authentication when using TLS for backup/restore.

Note:

Authentication is not supported for configuration or software file downloads.

Both Basic Server and Central File Server

Figure 5: Mode 4 Basic Server and Central File Server Mode



Use Case Scenario

Use the basic server and central file server combination mode to:

- Incorporate the features of Modes 1, 2, and 3,
- Offer peer to peer, highly distributed network configurations, and
- Mix and match Windows & Linux servers while remaining operating system independent.

Chapter 3: File Locations

Individual Server Descriptions

The basic central server architecture uses a standard Red Hat Linux or Windows server installation.

The sections in this chapter define the default locations for files. Avaya recommends that you use these default locations. Note also that the port numbers used are those used by default in the IP Telephone download process. The port numbers can be reassigned as needed for testing or another use.

The *Avaya one-X™ Deskphone Edition for 9600 Series IP Telephones Administrator Guide* defines settings for the actual 9600 Series IP Telephone data files. The *4600 Series IP Telephone LAN Administrator Guide* defines settings for the actual 4600 Series IP Telephone data files. The IP Telephone File Server Application servers replace all the equivalent servers described in these administrator guides. By providing a multi threaded approach, the application leverages that IP telephones can be programmed to search multiple servers/protocols on boot up. This leverage can provide more resilience or define more granular file delivery to the telephones.

TFTP Server with Default Port 69

The TFTP server function delivers configuration and firmware files to 4600 Series IP Telephones. These settings and files are normally stored in the **TFTPdata** directory of the application hierarchy by default:

For Linux: /opt/ecs/mvuser/MV_IPTel/data/TFTPdata

For Windows: c:/Program Files/Avaya/MV_IPTel/data/TFTPdata

Devices can access this directory with the TFTP protocol to retrieve the appropriate files.

Note:

TFTP is exactly that – trivial and not robust. See Improving TFTP Reliability on page 40 for options to improve download reliability by limiting connected clients by number or IP address and to allow notional “timed sessions.”

FTP Server Using Default Ports 21/20

The FTP server backs up individual 4600 Series IP Telephone user data such as screen settings and speed dial data. The data is stored in “data” directory by default:

For Linux: /opt/ecs/mvuser/MV_IPTel/data/FTPdata

For Windows: c:/Program Files/Avaya/MV_IPTel/data/FTPdata

Configuring 4600 Series IP Telephones for FTP backup/restore operation requires providing the FTP server IP address and optionally, a directory path, either in the settings file or by user input. Also, for additional security, the user must input a user name and a password. The Avaya IP Telephone File Server Application reduces the administration requirement to supplying the FTP address only, which should be downloaded from the settings file. Instead of requiring individual directories, user names and passwords, the FTP server interrogates the telephone during a backup or restore for its extension number using SNMP. The obvious prerequisite is that the user has logged in using their IP telephone credentials, which is mandatory to access these functions.

Note:

To simplify administration, only the FTP server address needs to be specified in the IP telephone settings file.

Note:

Remember to program the IP Telephone File Server Application server address as a valid SNMP query source in the 46xxsettings.txt file.

The user data is stored and restored by combination of phone extension and phone type retrieved in the SNMP query. Consequently, users can store personal settings and retrieve them in a hot-desking environment, since the data is associated with their personal extension number and not the physical phone instrument.

The Avaya IP Telephone File Server Application mode of operation provides additional security, since only an IP telephone responding correctly to the SNMP challenge during the storage process can either store or retrieve files. This allows the use of simple anonymous FTP logins, for example, not setting user name/password in each phone, to vastly simplify user administration.

The FTP server also has a Super User mode to allow storage of application files. Access is limited only to the IP Telephone File Server Application subdirectories. This provides an alternative method of storing files on the server.

Avaya recommends using a specialist FTP client program such as Cute FTP or WS_FTP to access the IP Telephone File Server Application FTP server. The more explicit messages provided regarding log in and change of working directory status provide better user feedback than trying to use FTP mode in something like Windows File Explorer. Use of Explorer is currently supported for the Windows server version only.

The servers have comprehensive logging capabilities for tracing problems and can be remotely interrogated for status using a built-in HTTP status server.

Note:

Remember to program the 46xxsettings.txt file to allow SNMP query access to the phone from the File Server Application server. The IP Telephone File Server Application invokes NetSNMP on Linux to retrieve this data from the telephone.

Accessing the File Server Application from an FTP Client Program

Set your FTP client to use the current Super User name and password. Set the operating system type of the File Server application server to which you are you are connecting. By default you are placed in what is considered the “home” directory, for example, /**opt/ecs/mvuser/MV_IPTel/** for Linux or **c:\Program Files\Avaya\MV_IPTel** for Windows.

Successful connection provides the directory list. However, since you can work only in the File Server application subdirectories, explicitly change (CWD) to the directory you want to use after successful login. For example, **CWD data/FTPdata** changes to the Linux subdirectories or **CWD data\FTPdata** changes to the Windows subdirectories.

HTTP/HTTPS Servers with Default Ports 411/81/80

Both 9600 Series IP Telephones and 4600 Series IP Telephones support HTTP, and by default, use the following port assignments:

- Port 411 for TLS download of configuration files from the IP addresses listed in the system parameter TLSSRVR.
- If configuration file download succeeds, the telephones then attempt to obtain firmware files from Port 81 from the IP addresses listed in the system parameter HTTPSRVR.
- If the attempt to get firmware files fails, the telephones then use Port 80.
- If the system parameter TLSSRVR is null, Port 80 is used with the IP addresses listed in the system parameter HTTPSRVR for all file downloads.

The HTTP server function delivers configuration and firmware files to IP telephones. By default, these settings are stored in the HTTPdata directory of the application hierarchy:

For Linux: /opt/ecs/mvuser/MV_IPTel/data/HTTPdata

For Windows: c:/Program Files/Avaya/MV_IPTel/data/HTTPdata

A secure download option is also available with an HTTPS server – which uses these defaults instead:

For Linux: /opt/ecs/mvuser/MV_IPTel/data/HTTPSdata

For Windows: c:/Program Files/Avaya/MV_IPTel/data/HTTPSdata

For secure operation, the File Server application supports TLS and specifically, the authenticate only mode used by Avaya IP Telephones. In addition, the older SSLV3 and SSLV2 protocols are supported but not enabled by default.

Chapter 4: Installing the Linux Server

Use Red Hat Package Manager (RPM) to install the application on Red Hat Linux Enterprise Linux V3.0, Update 4 or 5.

Note:

The application has also been tested on Red Hat versions 8 and 9, and Fedora Versions 1 and 2. Issues might arise with older Red Hat versions like 2.1.

A standard server installation is used but the secure FTP mode uses “**net-snmp**”. Both “**net-snmp**” and “**net-snmp-utils**” rpms must be installed.

The File Server application software is supplied as an installation script with the name:

- **MV_IPTel_Install.sh** for the core package, or
- **MV_IPTel_Full.sh** for the version containing HeartBeat.

To install the application:

1. Log in as root.
2. Copy the script to a convenient directory like /tmp.
3. Enter **`chmod + 760 MV_IPTel_Install.sh`** or **`MV_IPTel.Full.sh`**, as applicable.
4. Enter **`./MV_IPTel_Install.sh or MV_IPTel.Full.sh`**, as applicable.

The script installs all the binaries and necessary files including adding the daemon to the startup service list. During the installation, you are prompted to accept the license terms and decide whether to install HeartBeat. The script uses an embedded RPM package.

Note:

The Linux daemons run with root privileges, but the home directory is under the **mvuser** account. The data subdirectories are the only directories accessible to the IP telephones. The IP telephones are verified with the SNMP challenge by default. The Update capability automatically expands and installs a new release of IP Telephone firmware downloaded from the Avaya Web site, http://www.avaya.com/support. Update any of the data files in the Linux directory with any Linux utility valid for the **mvuser** login. Avaya recommends using secure sockets applications, for example, scp or sftp to modify files on the server. Alternatively you can try the IP Telephone File Server Application update service. The update capability allows a new release of IP telephone firmware downloaded from the Avaya support Web site to be automatically expanded and installed in the correct directories. It is up to the system administrator to decide whether a small change is required or a complete new load of IP telephone software is to be made available.

Linux Directory Structure

Verify that the installation has created the directory structure shown in Table 1.

Table 1: Linux Directory Structure

Directory	Application & function
/opt/ecs/mvuser/MV_IPTel/bin	**MV_IPTelD**: Runs FTP, HTTP, HTTPS and FTPS servers together with a built-in HTTP status server. **MV_IPTel.ini**: Text file containing configuration options. **Avaya_License**: Text file containing the license terms.
/opt/ecs/mvuser/MV_IPTel/FTPdata	Storage for 4600 Series IP Telephone text files. Archives are stored by Month in gzip files in sub directories.
/opt/ecs/mvuser/MV_IPTel/TFTPdata	Storage for 4600 Series IP Telephone application files and the setting for TFTP download.
/opt/ecs/mvuser/MV_IPTel/data/HTTPdata	Storage for application files and the setting for HTTP download.
/opt/ecs/mvuser/MV_IPTel/data/HTTPSdata	Storage for application files and the setting for HTTPS download.
/opt/ecs/mvuser/MV_IPTel/data/Scan	Storage for MV_Scan_Console output.
/opt/ecs/mvuser/MV_IPTel/certs	MV_IPTel PEM X509 certificate & private key files.
/opt/ecs/mvuser/MV_IPTel/log	MV_IPTel Log files – limited by size.
/opt/ecs/mvuser/MV_WatchDog/bin	**MV_WatchDogD**: Runs a watchdog service for all enabled servers. **MV_WatchDog.ini**: Text file containing configuration options.
/opt/ecs/mvuser/MV_WatchDog/data	Stores the “check files” used by MV_WatchDog.
/opt/ecs/mvuser/MV_WatchDog/log	MV_WatchDog Log files – limited by size.
/opt/ecs/mvuser/MV_Manager/bin	**MV_ManagerD**: Runs the Web management service for all enabled servers. **MV_Manager.ini**: Text file containing configuration options.
/opt/ecs/mvuser/MV_Manager/log	MV_Manager Log files – limited by size.

MV_IPTel .ini Configuration File

Avaya supplies a standard default set of options to run the IP Telephone File Server Application. Very little needs to be configured on the Linux sever for initial operation. A single configuration file **/opt/ecs/mvuser/MV_IPTel/bin/MV_IPTel.ini** maintains any changes to standard defaults.

Note:

If you make configuration changes, restart the daemons to activate the new settings.

The .ini file is self-documented. A sample format follows:

Sample .ini File Format

```
[Settings]
  #=====================================================================
  # Set detailed log for additional debugging info
  DetailedLog=1
  RunStatus=1
  StatusPort=6090
  StatusRefresh=10
  Archive=1
  # Sets the location of the MV_FTP log file
  LogFile=/opt/ecs/mvuser/MV_IPTel/log/MV_IPTel.log
  #=====================================================================
  Version=0.9 Build 4 Created July 11 2004 14:00
  ServerName=Unknown
```

```
[FTP]
  #=========================================================================
  # set the FTP server active
  RunFTP=1
  # defines the FTP control port
  FTPPort=21
  # defines the FTP data port
  FTPDataPort=20
  # Sets the location of the FTP data directory to catch terminal
    backups
  FTPDir=/opt/ecs/mvuser/MV_IPTel/data/FTPdata
  # FTP Timeout (secs)
  FTP_TimeOut=5
  # Enable SuperUser
  EnableSU=1
  # set the SuperUser Name
  SUUserName=mvuser
  # set the SuperUser Password
  SUPassword=Avaya
  #
  #=========================================================================
[FTPS]
  #=========================================================================
  # set the FTPS server active
  RunFTPS=0
  # defines the FTP control port
  FTPPort=990
  # defines the FTP data port
  FTPDataPort=889
  #=========================================================================
```

```
[TFTP]
  #=========================================================================
  # set the Trivial FTP server active
  RunTrivialFTP=1
  # defines the Trivial FTP port
  TrivialFTPPort=69
  # Sets the location of the TFTP data directory for terminal
    downloads
  TFTPDir=/opt/ecs/mvuser/MV_IPTel/data/TFTPdata
  #=========================================================================
[HTTP]
  #=========================================================================
  # set the HTTP download server active
  RunHTTP=1
  # defines the HTTP download port
  HTTPPort=81
  # Sets the location of the HTTP data directory for downloads
  HTTPDir=/opt/ecs/mvuser/MV_IPTel/data/HTTPdata
  #=========================================================================
[HTTPS]
  #=========================================================================
  # set the HTTPS download server active
  RunHTTPS=0
  # defines the HTTPS download port
  HTTPSPort=411
  # Sets the location of the HTTPS data directory for downloads
  HTTPSDir=/opt/ecs/mvuser/MV_IPTel/data/HTTPSdata
  # Sets the location of the CertFile
  CertFile=/opt/ecs/mvuser/MV_IPTel/certs/IPTelcert.pem
  # Sets the location of the KeyFile
  KeyFile=/opt/ecs/mvuser/MV_IPTel/certs/IPTelkey.pem
  # Use Client Authorization
  ClientAuth=0
  # narrow config for Avaya IPTel (TLSV1 using RSA_NULL_SHA)
  IPTel=0
  # sets the SSL variants if not Avaya IPtel (IPTel=0)
  SSLV2=0
  SSLV3=0
  TLSV1=1
  UseProxy=0
  ProxyAddr=simon.avaya.com
  ProxyPort=9000
  #=========================================================================
```

```
[BACKUP_SERVERS]
  #========================================================================
  # Run as FileServer for Backup & Update requests - Note this
    process uses HTTPS
  FileServer=0
  # sets whether to download Firmware updates from the primary/
    secondary file servers
  RequestUpdates=0
  # sets whether to upload FTP files to the primary/secondary file
    servers
  RequestBackup=0
  # Enable use of the Primary file server
  UsePrimarySvr=0
  # Primary file server IP address ( or resolvable DNS)
  PrimaryIP=192.168.0.13
  # Enable use of the Secondary file server
  UseSecondarySvr=0
  # Secondary file server IP address ( or resolvable DNS)
  SecondaryIP=192.168.0.10
  # Sets the update interval for Backups & updates ; 1 = min; 2
    =hour, 3=day, 4 =month
  UpdateInterval=2
  #Send FTP backup to the customer sever
  CustomFTP=1
  # FTP backup directory customer sever
  CustomFTPDir=home/mvuser/backup
  # FTP backup directory user login name
  CustomFTPUName=tom
  # FTP backup directory user password
  CustomFTPPwd=jerry
  # Enable CDR Backup - enable=1 on both File Server & Client
  CDRBackup=0
  # Enable BCMS Backup - enable=1 on both File Server & Client
  BCMSBackup=0
  # Retain CDR / BCMS copy data for x days ( Receiver always + 1
    week)
  RetainDays=7.0
  #========================================================================
```

```
[SNMP]
  #================================================================
  #
  # Validate FTP store with SNMP check
  UseSNMP=1
  # In case the SNMPGET syntax changes you can redefine the commands
  # Uncomment the relevant line to override the internal command
  #the syntax is "Command + IPADDR + ExtObj + Awk
  # the IPADRR is derived from the connection
  # Note there are relavant spaces at the start/end of the component
    - omit and it will fail
  #Command=/usr/bin/snmpget
  #Params= -v2c -cpublic
  #ExtObject=.1.3.6.1.4.1.6889.2.69.1.4.9.0
  #TypeObject=.1.3.6.1.4.1.6889.2.69.1.1.2.0
  #Awk=| awk -F \" '' {print $2 } ''
  #================================================================
```

Chapter 5: Installing the Windows Server

Use the Windows installer application to install the IP Telephone File Server Application on Windows NT/2000 or XP servers. Unlike the Linux version, the Windows version of the application includes an SNMP query agent used for secure FTP.

The IP Telephone File Server Application software is supplied as an installation executable with the name **MV_IPTel_setup.exe**.

To install the application:

1. Log in as the administrative user on the server.
2. Copy the MV_IPTel_setup.exe file to a convenient installation directory such as c:\temp.
3. To install the software, double-click **MV_IPTel_setup.exe** file within the File Explorer.

All the binaries and necessary files are installed including adding the service to the startup service list. During the installation, you are prompted to accept the license terms and choose which type of installation you want to use:

- **Minimal** – only installs the File Server application server and a simple GUI manager.
- **Typical** – adds the Web administration.
- **Custom** – adds the WatchDog.

The GUI manager (**MV_Mgr.exe**) allows access to the Windows service controls and to check the status of or modify server options. The defaults allow the server to be automatically enabled.

Note:

The Windows services run with administrator privileges. However, the data in the subdirectories defined in Table 2 are the only files accessible to IP telephones after SNMP challenge verification. Should the user require full access to these data sub-directories, set the Super User flag to **1** in the FTP section of the MV_IPTel.ini file and restart the MV_IPTel server service.

Note:

The installation program allows the default location (c:\Program Files\Avaya\MV_IPTel) to be changed. Avaya does not recommend changing the default location.

Windows Directory Structure

Verify that the installation has installed the directory structure shown in Table 2.

Table 2: Windows Directory Structure

Directory	Application & function
C:\Program Files\Avaya\MV_IPTel\bin	**MV_IPTel_Services.exe**: Runs an FTP, HTTP, HTTPS (and FTPS) function together with a built-in HTTP status server. **MV_Mgr.exe**: A local GUI manager for the suite. **MV_IPTel.ini**: Text file containing configuration options. **Avaya_License**: Text file containing the license terms.
C:\Program Files\Avaya\MV_IPTel\data\FTPdata	Storage for IP phone text files. Archives are stored by Month in gzip files in subdirectories.
C:\Program Files\Avaya\MV_IPTel\data\TFTPdata	Storage for application files and the setting for TFTP download.
C:\Program Files\Avaya\MV_IPTel\data\HTTPdata	Storage for application files and the setting for HTTP download.
C:\Program Files\Avaya\MV_IPTel\data\HTTPSdata	Storage for application files and the setting for HTTPS download.
C:\Program Files\Avaya\MV_IPTel\data\Scan	Storage for MV_Scan_Console output.
C:\Program Files\Avaya\MV_IPTel\certs	MV_IPTel X509 Certificate and Private Key files – PEM format.
C:\Program Files\Avaya\MV_IPTel\log	MV_IPTel Log files – limited by size.
C:\Program Files\Avaya \MV_WatchDog\bin	**MV_WatchDog_Service.exe**: Runs a watchdog service for all enabled servers. **MV_WatchDog.ini**: Text file containing configuration options.
C:\Program Files\Avaya MV_WatchDog\data	Stores the “Check files” used by MV_WatchDog.
C:\Program Files\Avaya \MV_WatchDog/log	MV_WatchDog Log files – limited by size.
C:\Program Files\Avaya MV_Manager\bin	**MV_Manager_Service.exe**: Runs the Web management service for all enabled servers. **MV_Manager.ini**: Text file containing configuration options.
C:\Program Files\Avaya \MV_Manager\log	MV_Manager Log files – limited by size.

MV_IPTel .ini Configuration File

Avaya supplies a default set of options to run the File Server application. Minimal data like the server name and your IP configuration must be configured on the Linux sever for initial operation. A single configuration file, **C:\Program Files\Avaya\MV_IPTel\bin\MV_IPTel.ini** maintains any changes to standard defaults. The .ini file is self documented. A sample format follows:

Note:

If you make configuration changes, use the control panel to restart the service to activate the new settings.

Sample .ini File Format

```
[Settings]
  #1=====================================================================
  #2= Set detailed log for additional debugging info
  DetailedLog=1
  RunStatus=1
  StatusPort=6090
  StatusRefresh=10
  Archive=0
  #3= Sets the location of the MV_IPTel log file
  LogFile=c:\Program Files\Avaya\MV_IPTel\log\MV_IPTel.log
  #4=====================================================================
  ServerName=SMW_VAIO
  IPAddress=192.168.0.10
```

```
[FTP]
  #5=====================================================================
  #6= set the FTP server active
  RunFTP=1
  #7= defines the FTP control port
  FTPPort=21
  #8= defines the FTP data port
  FTPDataPort=20
  #9= Sets the location of the FTP data directory to catch terminal
    backups
  FTPDir=c:\Program Files\Avaya\MV_IPTel\data\FTPdata
  #10= FTP Timeout (secs)
  FTP_ TimeOut=5
  #11= Enable the FTP Super User access to the MV_FTP data
    directories
  EnableSU=0
  #12= set the FTP Super User Name - only valid when EnableSU = 1
  SUUserName=mvuser
  #13= set the FTP Super User Password -only valid when EnableSU = 1
  SUPassword=Avaya
[FTPS]
  #15=====================================================================
  #16= set the FTP server active
  RunFTPS=1
  #17= defines the FTP control port
  FTPSPort=990
  #18= defines the FTP data port
  FTPSDataPort=889
  #19=====================================================================
```

```
[TFTP]
  #20=================================================================
  #21= set the Trivial FTP server active
  RunTrivialFTP=1
  #22= defines the Trivial FTP port
  TrivialFTPPort=69
  #23= Sets the location of the TFTP data directory for terminal
    downloads
  TFTPDir=c:\Program Files\Avaya\MV_IPTel\data\TFTPdata
  #24=================================================================
[HTTP]
  #25=================================================================
  #26= set the HTTP download server active
  RunHTTP=1
  #27= defines the HTTP download port
  HTTPPort=81
  #28= Sets the location of the HTTP data directory for downloads
  HTTPDir=c:\Program Files\Avaya\MV_IPTel\data\HTTPdata
  #29=================================================================
[HTTPS]
  #30=================================================================
  #31= set the HTTPS download server active
  RunHTTPS=1
  #32= defines the HTTPS download port
  HTTPSPort=411
  #33= Sets the location of the HTTPS data directory for downloads
  HTTPSDir=c:\Program Files\Avaya\MV_IPTel\data\HTTPSdata
  #34= Sets the location of the CertFile
  CertFile=c:\Program Files\Avaya\MV_IPTel\certs\IPTelcert.pem
  #35= Sets the location of the KeyFile
  KeyFile=c:\Program Files\Avaya\MV_IPTel\certs\IPTelkey.pem
  #36= Use Client Authorization
  ClientAuth=0
  #37= narrow config for Avaya IPTel (TLSV1 using RSA_NULL_SHA)
  IPTel=0
  #38= set SSL variants
  SSLV2=0
  SSLV3=0
  TLSV1=1
  #39=================================================================
```

```
[BACKUP_SERVERS]
  #40=====================================================================
  #41= Run as FileServer for Backup & Update requests - Note this
    process uses HTTPS
  FileServer=1
  #42= sets whether to download Firmware updates from the Primary/
    Secondary File servers
  RequestUpdates=1
  #43= sets whether to upload FTP files to the Primary/Secondary
    file servers
  RequestBackup=1
  #44= Enable use of the Primary file server
  UsePrimarySvr=1
  #45= Primary file server IP address (or resolvable DNS)
  PrimaryIP=192.168.0.10
  #46= Enable use of the Secondary file server
  UseSecondarySvr=0
  #47= Secondary file server IP address (or resolvable DNS)
  SecondaryIP=192.168.0.10
  # Sets the update interval for Backups & updates: 1 = min, 2
    =hour, 3=day, 4 =month
  UpdateInterval=2
  #Send FTP backup to the customer sever
  CustomFTP=1
  # FTP backup directory customer sever
  CustomFTPDir=home/mvuser/backup
  # FTP backup directory user login name
  CustomFTPUName=tom
  # FTP backup directory user password
  CustomFTPPwd=jerry
  # Enable CDR Backup - enable=1 on both File Server & Client
  CDRBackup=0
  # Enable BCMS Backup - enable=1 on both File Server & Client
  BCMSBackup=0
  # Retain CDR / BCMS copy data for x days (Receiver always + 1
    week)
  RetainDays=7.0
  #48=====================================================================
```

```
[SNMP]
  #49=================================================================
  #50 Use SNMP to verify telephone identity
  UseSNMP=1
  #51= In case the SNMPGET syntax changes you can redefine the
    commands
  #52= Uncomment the relevant line to override the internal command
  #53= the syntax is "Command + IPADDR + ExtObj
  #54= the IPADRR is derived from the connection
  #55= Note there are relevant spaces at the start/end of the
    component - omit and it will fail
  #56= Command=mvsnmpget.exe
  #57= Params= -v2c -cpublic
  #58= ExtObject=.1.3.6.1.4.1.6889.2.69.1.4.9.0
  #59= TypeObject=.1.3.6.1.4.1.6889.2.69.1.1.2.0
  #60=================================================================
```

Chapter 6: Optimizing the Server

This chapter describes the options available to improve the robustness of the server. The options are:

- WatchDog Server
- HeartBeat redundant solution for Linux
- Firmware Update service
- IP Access Control List
- FTP File Server Backup
- CDR/BCMS Data copy
- TFTP “Session Time” Options

WatchDog Operation

The WatchDog server is a separate application designed to monitor the health of the IP Telephone File Server Application main server. The WatchDog application senses which servers the IP Telephone File Server Application is running and periodically requests download files. If the download files fail to arrive, WatchDog triggers a recovery script to reset the IP Telephone File Server Application. Use the Web manager to run the server and modify the INI files as needed. Figure 6 shows the .ini file content.

Figure 6: WatchDog .ini File Content

```
[Settings]
  #================================================================
  Version=0.9 Build 1 Created July 11 2004 14:00
  # Address of host to monitor (normally local)
  HostAddress=127.0.0.1
  # Time in seconds between file downloads (min 1)
  WatchDogTimer=60
  # Total number of failed requests to trigger watchdog
  FailureLimit=5
  StatusPort=6091
  StatusRefresh=10
  # Sets the location of the MV_Watchdog log file
  LogFile=c:\Program Files\Avaya\MV_Watchdog\log\MV_Watchdog.log
  #================================================================
```

By default, the WatchDog requests files every 60 seconds (**WatchDogTimer)**. If the number of missed files climbs to the **FailureLimit** setting, a reset takes place within approximately one minute. The WatchDog also counts failures back down to **0**, if successful downloads recur before the timer expires.

If a failure occurs, WatchDog executes the following default script files if found:

Linux: **/opt/ecs/MV_WatchDog/bin/mv_iptel_alert.sh** (this script file restarts the daemons).

Windows: **C:\Program Files\Avaya\MV_WatchDog\bin\mv_iptel_alert.bat** (this script file stops and restarts the services).

You can modify the scripts to execute any valid commands on the host system, such as setting SNMP traps or sending e-mail.

HeartBeat Redundant Server

HeartBeat is a High Availability Linux solution designed to provide servers with hardware redundancy. HeartBeat provides a primary/hot standby server pairing which can share a single IP address to the external world. HeartBeat continually checks its partner server over a private link. In the event of failure, HeartBeat allows the “services” it supports to become active on the standby server until the primary server recovers.

HeartBeat is used in conjunction with MV_WatchDog, which provides application level monitoring.

Detailed information on HeartBeat setup is provided in Chapter 9: HeartBeat. The installation script provided does all the setup, provided the defaults are acceptable.

Tthe Linux-HA project provides HeartBeat. For more information, see:

http://www.Linux-HA.org

Updating Firmware

The Avaya IP Telephone File Server Application server can act in one of 4 modes:

- As an TFTP/FTP/HTTP/HTTPS server, called the basic Avaya IP Telephone File Server Application server to support the IP telephones.

  Note:

  TFTP and FTP servers apply only to 4600 Series IP Telephones.

- As a basic Avaya IP Telephone File Server Application server plus as a client to pass data to a central fileserver.
- As a basic Avaya IP Telephone File Server Application server.
- As both file server and client server.

These various options can be selected within the configuration tools provided.

With the exception of the basic mode, the IP Telephone File Server Application software acts as either the client or server to provide simple file server firmware download. The IP Telephone File Server Application uses the HTTPS server for secure transmission using TLS.

Note:

The IP Telephone File Server Application uses RSA, AES256 and SHA security standards as defaults.

Acting as a client, the IP Telephone File Server Application server requests firmware updates from a central file server. When set as the client and optioned for firmware updates, the IP Telephone File Server Application sever periodically requests a list of any new firmware from the primary/secondary file server, in that order.

A list of all available firmware or other files stored in the file server **Updates** directory is delivered to the IP Telephone File Server Application server acting as a client. The file server application checks the files in its own Updates directory and requests all files it does not have using HTTPS GET requests.

On successfully receiving new files, the server unzips them and places copies in each of the **TFTPdata** (46xx only), **HTTPdata**, and **HTTPSdata** subdirectories. The new files are then available for telephone download.

To use this feature, assign the following configuration items:

1. Choose the **mode** for each IP Telephone File Server Application server (client or file server).
2. Enable the **Firmware distribution** option in the **INI** setting.
3. Assign a primary and optional secondary **file server address** in each IP Telephone File Server Application client to look for new data.
4. Select the **frequency of update**. The default is “hourly.”

To check the backup operation, place a file in the **Updates** subdirectory of the file server. Verify that it is received and stored in the **Updates** directory of the IP Telephone File Server Application basic server. Ensure that zipped files have also been unzipped.

Note:

If you have both Linux and Windows file servers, it might be easier to ZIP file telephone archives only. Each format can, however, be unpacked in the local IP Telephone File Server Application server.

Note:

HTTPS proxy support is built in, which allows broader use over an extranet or the Internet. This mode is currently untested.

FTP File Server Backup Operation

The functions this section describes apply only to 4600 Series IP Telephones.

Backup Using the IP Telephone File Server Application as a File Server

This operation provides periodic back up of FTP data to a primary and/or secondary server running the file server application software. The data can be backed up every month/day/hour or minute, the latter used normally for testing.

The FTP data backup operation acts like the TFTP firmware distribution, but in reverse. The backup process works by the IP Telephone File Server Application building a list of the files in its **FTPdata** subdirectory and issuing a request to the file server to GET these files. The file server then issues multiple HTTPS “GET requests” back to the client to collect the backup list and all the files the list contains. Only new or newer files are downloaded and stored in the file server **BackUp** directory.

Automatic FTP data recovery is provided to the end user. When the telephone makes a request to the FTP server and that server is not there for some reason, the local IP Telephone File Server Application server will make a request to its designated Primary, then Secondary file server to see if the file is available there. If found, the file is downloaded and stored locally before being served to the local customer. This mode of operation can also help to deliver personal phone data to users who roam in a corporate network and log in elsewhere, provided a Uniform dial plan is used.

To use the backup feature, you must assign a number of configuration items:

1. Choose the mode for each given IP Telephone File Server Application server (client or file server).
2. Enable the **FTP Backup** option in the INI setting.
3. Assign a primary and optional secondary file server address in each IP Telephone File Server Application client.
4. Select the frequency of update. The default is hourly.

To check the backup operation, place a file in the **FTPdata** subdirectory and check that it is received and stored in the **Backup** directory of the designated file server.

Backup Using Automatic Archive

If the Archive flag is set in the INI file settings section, the IP Telephone File Server Application automatically backs up the data in the **FTPdata** directory. The archive works on a rolling seven day basis naming the files **Sun_FTParchive.zip**, **Mon_FTParchive.zip**, and so on.

Archiving is typically done every hour. A copy of the latest archive is placed in the **Backup** directory to allow it to be copied automatically to an IP Telephone File Server Application file server.

In addition, you can automatically FTP this backup file to another central FTP server – which could be anonymous or secure – and still avoid the administrative overhead of individual FTP user administration.

To use automatic archive:

1. In the MV_Manager, go to the **Advanced settings** page.
2. Set the **Customer FTP** Backup flag to enabled.
3. Enter the **Backup Directory** for the remote FTP server.
4. Enter the customer server **FTP UserName**.
5. Enter the customer server **FTP Password**.

Also ensure that the primary file server address is entered and enabled and/or a secondary file server address is enabled. Then choose the backup interval required.

Note:

It is not normally expected that the option outlined in this and the previous section would be active simultaneously, but can be within the administration limits currently set. If so, some backup activity will be ignored if offered to servers not set up to receive it.

To set up an alternative backup to a network storage unit, link the Backup directory to a network share file server or use another utility program to mirror the data.

Improving TFTP Reliability

TFTP can be problematic in busy networks. Two options are available to improve TFTP file download to 4600 Series IP Telephones:

- Set the **Maximum Client** limit (0 = unlimited) in the INI file (applies to all server modes), to control the number of individual telephones accessing the server, by storing a list of individual IP addresses. The server itself can handle hundreds of connections in parallel. The telephone, however, must download several files to complete an upgrade and may be blocked by others accessing the server. Setting a limit like 100 ensures that individual phones get more access to the server.
- Assign a **Client Session** time in seconds with 0 = unlimited. Setting a session time reserves a time slot on a server for that duration to allow file downloads to be completed. This parameter effectively triggers when the Maximum Client count limit is reached because until that point, all new requests from unique IP addresses are accepted.

IP Access Control Lists

When deploying new IP telephone software, it is likely that you will use a subset of the telephone population for testing.

The Avaya IP Telephone File Server Application server offers the Access control list concept to define which IP addresses are allowed access to its various files. Using IP Access Control lists, each HTTP and HTTPS server checks the data subdirectory (**TFTPdata**, **FTPdata**, **HTTPdata**, or **HTTPS data**) for a fixed name file prefixed with its type. For example:

"**TFTP_allow_IP.txt**" (46xx only)

"**FTP_allow_IP.txt**" (46xx only)

"**HTTP_allow_IP.txt**"

"**HTTPS_allow_IP.txt**"

Alternatively, you can place the same generic file "**allow_IP.txt**" in each directory.

Note:

The **xxx_allow_IP.txt** file itself can be automatically distributed as part of the TFTP download mechanism if placed in zipped format in the **Updates** directory of the fileserver. All the contents are unzipped and placed in the respective sub-directories. Alternatively, an administrator can use the FTP SuperUser mode to log into the remote IP Telephone File Server Application server and place access control lists in the given server directories.

If one of these files exists, it is used for each GET request to determine if the requester is listed. The server-specific file is used if it exists. Alternatively, the request uses the generic file.

The list format takes one of three styles:

- a full IP address, for example, **192.168.0.1**,
- a partial IP address, for example, **192.168.0**, or
- a range without spaces in the form **192.168.0.1-192.168.0.128**, for example.

Put each entry in a simple text file on a separate line.

Overriding Downloads

IP telephones that are reset unexpectedly might not access the appropriate upgrade and settings files. In this case, the telephones default to a mode which users can find unacceptable, such as having missing features or user-defined options.

Using either a protocol-specific access list or a general access list controls server access. You can also bypass these control mechanisms and force a default upgrade and settings back to the telephone. Doing so prevents the telephone from requesting more file downloads in these conditions.

The server checks for the existence of one of two files in the TFTP/HTTP/HTTPS directory

"**fxd_upgrade.scr**" and "**fxd_settings.txt**".

When requesting the normal **46xxsettings.scr** and **46xxsettings.txt** files and access would otherwise be denied, the Avaya IP Telephone File Server Application server delivers the "**fxd_xx…."** files instead.

These files are expected to be programmed with parameters suited to the failure mode and conducive to just restoring basic telephone operation quickly. File examples are in the "docs" subdirectory of the application.

Scanning IP Telephone Firmware

The File Server application Status Web page has a built-in scan function that checks the current status of IP telephone firmware. MV_Scan_Console invokes a sequential set of SNMP queries to a range of IP addresses and stores the results in both detail and summary format.

You can set the start and stop range. There is an option to set a maximum count of the telephones searched with a default = 1000.

To search a range of IP addresses, create a text file with the start and stop range. Place the text file in the "**Scan**" subdirectory. Enter the file name in the Web form. This method overrides the start and stop range settings.

The format is a list of ranges formatted with start and stop IP addresses separated by a space on separate lines. For example:

192.168.0.1 192.168.0.128

192.168.35.27 192.168.35.97

10.0.0.1 10.0.0.50

Note:

The Scan subdirectory of the application has examples of a parameter file.

To scan, press the **SCAN** button on the Web page. Because the scan might take some time, refresh the page to check for progress or completion.

Note:

Windows XP scanning might produce an invalid address range. When scanning an invalid network routing address, your Net-SNMP snmpget utility version can create a Windows XP error message. Although scanning continues automatically and there is no consequence to the error message, the message must be closed manually. Contact Avaya support if you encounter problems.

Copying CDR/BCMS Data

MV_CDR collects Communication Manager call detail records generated at the end of outbound calls. MV_BCMS collects Basic Call Measurement System records generated for inbound calls from Communication Manager to measure agents and splits/skills.

The IP Telephone File Server Application with MV_CDR and/or MV_BCMS can copy data securely between servers using the SSL/TLSV1 protocol. For example, CDR data might need to be captured and analyzed at both a branch office and a central site.

In this scenario, MV_CDR or MV_BCMS captures the raw data from the PBX. If the INI file **CopyToIPTel** flag is set, the import part of either application copies the captured data to the Avaya IP Telephone File Server Application CDRdata or BCMSdata subdirectories. To relay the data between sites, the Avaya IP Telephone File Server Application must be running on both servers with the HTTPS servers enabled.

At the sending end, the option **CDRBackup** or **BCMSBackup** must be set in the INI file together with:

- the backup frequency in minutes/hours/days format, and
- how many days to retain the data – **RetainDays**. The retention days are in units of 1 day and can be less than 1.0. For example, use “0.5” to represent 12 hours.

Data is automatically purged at both ends, with the receiving end allowing one additional week for the data to be processed.

The sending end must be programmed with the primary and, if needed, secondary file servers and with relevant DNS names or IP addresses set.

Figure 7: Sample .ini File Settings for CDR/BCMS

Sending end ini settings needed:

```
#=============================================
[HTTPS]
  RunHTTPS=1
  HTTPSPort=411
  TLSV1=1
[BACKUP_SERVERS]
  #=============================================
  CDRBackup=1
  BCMSBackup=1
  RetainDays=7.0
  UpdateInterval=2
  UsePrimarySvr=1
  PrimaryIP=192.168.0.13
  UseSecondarySvr=0
  SecondaryIP=192.168.0.11
```

Note: At the receiving end, the Avaya IP Telephone File Server Application must be set as a fileserver to receive the incoming requests.

Receiving end ini settings needed:

```
  #=============================================
[HTTPS]
  RunHTTPS=1
  HTTPSPort=411
  TLSV1=1
[BACKUP_SERVERS]
  #=============================================
  FileServer=1
  CDRBackup=1
  BCMSBackup=1
  RetainDays=7.0
  UpdateInterval=2
```

The Avaya IP Telephone File Server Application performs most of the data relay function. With the options sets as shown in Figure 7, the Avaya IP Telephone File Server Application scans the two **CDRBackup or BCMSBackup** directories. The Avaya IP Telephone File Server Application creates a list of the files to be transferred. The storing end requests the list of files to be relayed on a periodic basis. The files in the list are uploaded to an equivalent **CDRBackup or BCMSBackup** directory on the file server. The receiving end then copies the data to the MV_CDR and/or MV_BCMS application as if the data was collected locally.

Using this mechanism and the redundant database options of MV_CXDR or MV_BCMS, you can perform highly secure and distributed management data collection.

Chapter 7: Maintaining Operations and Troubleshooting

Maintaining Operations

Checking Linux Operation

The Avaya IP Telephone File Server Application runs as a daemon on a Linux server and is configured to start up automatically on bootup. The MV_Manager daemon provides a service to start and stop the other services over the Web but clearly must first be running.

The easiest option to manually start and stop the daemon is the Red Hat Admin GUI Services utility. Alternatively, you can control stopping and starting from the command line as root.

To manually start the daemon, use the **start** qualifier, for example:

/etc/init.d/mv_managerd start - the MV_Manager daemon
/etc/init.d/mv_ipteld start - the Avaya IP Telephone File Server Application daemon
/etc/init.d/mv_watchdogd start - the MV_WatchDog daemon

To manually stop the daemon, use the **stop** qualifier, for example:

/etc/init.d/mv_managerd stop
/etc/init.d/mv_ipteld stop
/etc/init.d/mv_watchdogd stop

Note:

Server maintenance and any required data backup is the responsibility of the user. For more information, see the Red Hat documentation. Alternatively, use the automated backup service.

Checking Windows Operation

The Avaya IP Telephone File Server Application runs as a service on a Windows server and is configured to automatically start on power up. Therefore, no default administration is needed.

Using the MV_Mgr graphical utility you can change settings and control the services. An icon for this utility was installed during the Avaya IP Telephone File Server Application setup.

To manually check the service installation status and start the service, use the **Services** utility in the Control Panel administration GUI. Select **Control Panel, Administrative Tools**, then **Services**. Depending on the Windows version, menus allow for the Service to be started, restarted, paused, and stopped.

Note:

Server maintenance and any required data backup is the responsibility of the user. For more information, see the Windows Backup documentation.

Checking Application Status

To verify that the Avaya IP Telephone File Server Application daemon is running properly, access a built-in HTTP server that runs within each application. You can access the HTTP server either locally or remotely as follows:

- To access the MV_Manager server, start a Web browser and type:

  `http://server_address:6099`.

  Or, on the Avaya IP Telephone File Server Application server itself, type:

  `http://localhost:6099`.

  Note:

  The default Super User name and password are **mvuser** and **Avaya**.

- To access the Avaya IP Telephone File Server Application daemon http status server, start a Web browser and type

  `http://server_address:6090`.

  Or, on the Avaya IP Telephone File Server Application server itself, type:

  `http://localhost:6090`.

- To access the MV_WatchDog daemon http status server, start a Web browser and type

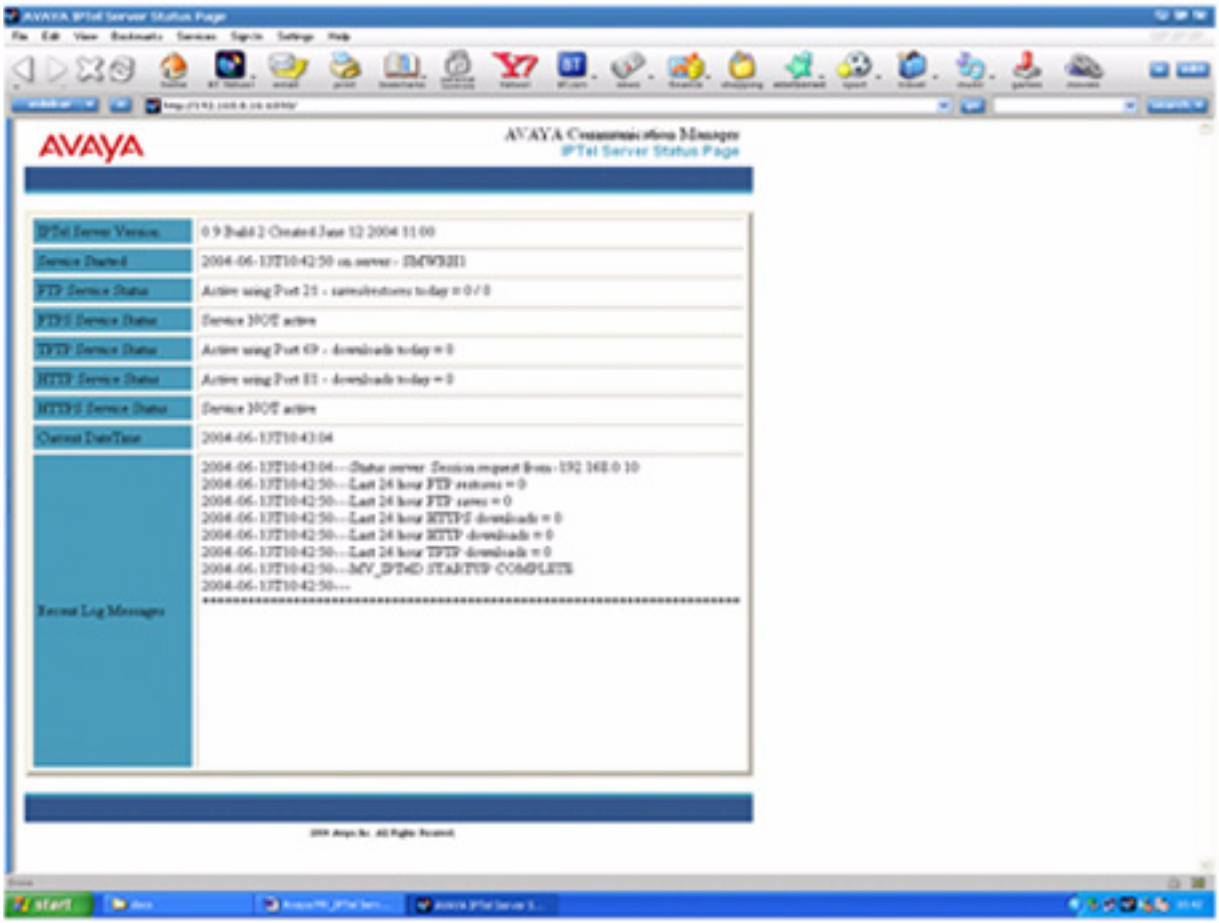
  `http://server_address:6091`.

  Or, on the Avaya IP Telephone File Server Application server itself, type:

  `http://localhost:6091`.

An example of the IPTel Status Page follows:

Figure 8: IPTel Status Page



Troubleshooting

Most problems with the Avaya IP Telephone File Server Application tools are usually configuration- or network-related. A comprehensive set of log files residing in the relevant **log** directory can help pinpoint any configuration mismatches.

If you want more detailed logging, set the **detail log flag** to **1** in the MV_IPTel.ini file. Although detailed logging increases output data significantly, it provides a good level of detail for all activities.

Use these steps to help troubleshoot problems:

1. Is the daemon you are trying to access running? First verify that the Avaya IP Telephone File Server Application server is running by launching a Web browser and pointing it to the built-in HTTP server. By default, these run on port 6090 for the Avaya IP Telephone File Server Application daemon, 6091 for MV_WatchDog and 6099 for MV_Manager. For more information, see Checking Application Status. Check that the port hasn't been changed. For Linux, use the **ps –ef | grep MV** Linux command or the graphical Services tool in Gnome or KDE. For Windows, check the Service list in the control panel.
2. For Linux, ensure that old versions are not running in parallel. Check using **ps –ef | grep MV**. Old versions might run in parallel if the daemons are started and stopped without using the scripts provided.
3. Are the standard file locations being used? If not, is the INI file set correctly? Any changes require a server restart to become effective. Use the tools available to check the settings in detail.

   Note:

   The current default configuration does not activate either the HTTPS or FTPS servers. At the time of release these functions were unavailable in the IP telephones.
4. If SNMP is not working, ensure that the File Server application server is referenced in the 46xx settings file for SNMP. For more information, see the *Avaya one-X™ Deskphone Edition for 9600 Series IP Telephones Administrator Guide* or the *4600 Series IP Telephone Administrator Guide* as applicable, both available on the Avaya support Web site http://www.avaya.com/support. Ensure that the NetSNMP package is available on Linux by issuing an **`SNMPGET`** command. For more information, see the SNMP section of the INI file. For example:

   ```
   smnpget -v2c -cpublic  phone-ip-addr
     .1.3.6.1.4.1.6889.2.69.1.4.9.0
   ```

   Ensure that the **`get`** command returns a valid extension for the IP address used.
5. Ensure that the file system has the correct permissions, particularly on a Linux server. Windows defaults are usually acceptable. Permissions are installed correctly by default, but can be changed manually.

6. Use the built-in facilities/logging of the File Server application to verify whether the HTTP functions are working. The watchdog provides a comprehensive check.
7. Ensure that the server has the correct networking setup and that any firewall functions are not preventing connection to the File Server application server. Ensure that you can access the server over the network. Do this by launching a browser and accessing the HTTP status server, as covered in Checking Application Status. Test files are installed in the HTTP servers.

   For HTTP: http://hostname:81/test.html and http://hostname:81/test.zip

   For HTTPS: https://hostname:4111/test.html and https://hostname:411/test.zip

   Note:

   For HTTPS checks, ensure that you select SSLV2 + SSLV3 as options for this test. Many browsers do not support the security algorithms used in the Avaya IP Telephone File Server Application for TLSV1 and SSLV3.

Chapter 8: DHCP Server Administration

Avaya IP Telephones are usually configured remotely to enable downloading the appropriate configuration files. Normal, dynamically assigned IP configurations need a DHCP server to allocate IP addresses to the telephones, as opposed to static assignment. The DHCP server can either be a separate Windows/Linux server or be configured on the same server as the Avaya IP Telephone File Server Application.

The DHCP server also allows the HTTP server address, meaning the IP Telephone File Server Application server, to be set for the telephones.

Important:

The most important setting is to ensure that the telephone knows the correct IP address of the HTTP settings file download from the IP Telephone File Server Application server.

For information about DHCP setup and for general administrative information, see the *Avaya one-X™ Deskphone Edition for 9600 Series IP Telephones Administrator Guide* or the *4600 Series IP Telephone Administrator Guide* as applicable. The guides are available on the Avaya support site http://avaya.com/support.

For more information about setting up the DHCP server on RedHat Linux, see the RedHat manual by typing **man dhcp-options** in a command shell window.

Essentially you need to:

1. Configure the **/etc/dhcpd.conf** file to the required settings for your specific IP address assignments. A text editor like KWrite can be helpful.
2. Start the **dhcpd** daemon and ensure that it starts automatically on reboot.

Here is a simple example of a **/etc/dhcpd.conf** file:

```
# A simple config file
default-lease-time 720;
max-lease-time 86400;
option subnet-mask 255.255.0.0;
option broadcast-address 192.168.255.255;
option routers 192.168.0.254;
option domain-name-servers 192.168.0.1, 192.168.0.2;
option tftp-server-name “MV_Server1”;
range 192.168.0.10 192.168.0.253
```

To ensure that the dhcpd daemon runs on startup, access the Service Configuration application from the GUI. Select **Menu**, **System Settings**, **Server Settings**, and **Services**. Verify that the **“dhcpd”** service is checked.

Chapter 9: HeartBeat

This section describes the high availability Linux application called “HeartBeat” in more detail. The automated installation script creates all the defaults covered in this chapter.

How HeartBeat Works

You can configure the Avaya IP Telephone File Server Application in a high availability mode using two servers. The application uses a combination of the HeartBeat High Availability Linux software and the RSync utility.

HeartBeat allows two servers to monitor the health of each other over a private Ethernet or serial link. HeartBeat manages a single, publicly available IP address for the Avaya IP Telephone File Server Application server to share. RSync is a sophisticated file comparison/ duplication utility. In this case, RSync mirrors the data files between the two servers.

Note:

To control a High Availability cluster you only use the HeartBeat daemon control. Do not manually start or stop the Avaya IP Telephone File Server Application or MV_WatchDog daemons. HeartBeat does this for you as needed.

The fundamental requirements are:

- two identical Intel-based servers with dual Ethernet connections.
- RedHat Linux Enterprise Server Version 3, Update 4 0r 5.
- Avaya IP Telephone File Server Application server software.
- HeartBeat software, available at http://linux-ha.org.
- RSync software, available at http://samba.org/rsync. The RSync software is also available as part of the Linux distribution.

Note:

The MV_IPTel_Install.sh is a self-extracting script which contains all the components needed beyond the Red Hat OS installation.

For additional redundancy, the servers benefit from additional options such as hot swap power supplies and software or hardware based RAID functions.

For more information about hardware setup options, see the HeartBeat documentation.

Our application uses a crossover Ethernet cable connected between the two servers for the basic HeartBeat function using the second Ethernet adaptors.

Important:

The following instructions assume the first server (EY-FTP1) is configured with Ethernet Address 10.0.0.1 and the second server (EY-FTP2) is configured with Ethernet Address 10.0.0.2. If your installation differs, change the configuration files accordingly. Since this Ethernet segment is private between the two servers, there should not be many circumstances where you need to change this configuration.

MV_IPTelD Considerations

MV_IPTelD is designed for use in a High Availability Cluster environment. However, there are several considerations for appropriate operation.

Basic Assumptions for Easy Installation

The following basic configuration assumptions apply for easy installation:

- The servers use private IP addresses **10.0.0.1** and **10.0.0.2** for monitoring.
- The private monitoring addresses are assigned to "**eth1**" interface in both servers.
- All default installation directories are used in the Linux Red Hat installation.
- The HeartBeat link is a simple Ethernet Cross Over cable linking the servers on their respective **eth1** interfaces.

Important:

If you change this basic configuration, ensure that you know how to modify the configuration files properly for HeartBeat operation.

Disabling Auto-Start of the Daemon

By default, installing MV_IPTelD causes the daemon to start on a restart of Linux. This start and restart are usually desirable since no manual intervention is required if the Linux server is restarted for whatever reason. When operating in a High Availability mode, the HeartBeat application has to decide when and if this daemon should start or stop. Therefore, it is essential that this daemon is not enabled for auto-start. The MV_IPTel_Install.sh script does this automatically.

Use the “services” application and deselect the MV_IPTelD daemon from starting automatically to verify that auto-start is disabled.

SNMP Configuration of IP Telephones

IP Hardphone software supports configuration parameters allowing control of SNMP servers that can query the IP Hardphones. In a single IP Telephone File Server Application server environment, only the IP address for this server needs to be enabled. But for a High Availability cluster, both real IP addresses and the shared virtual IP address must be enabled.

HeartBeat Installation and Configuration

The HeartBeat application used for this document was 1.0.4, in which the HeartBeat software makes use of the external library files PILS and STONITH. Because of HeartBeat RPM dependencies, install all three RPM’s with a single command line such as:

```
rpm –Uvh heartbeat*.rpm
```

Three small configuration files control the HeartBeat application operation and must reside in the “**/etc/ha.d**” directory.

Configuration File: ha.cf

The **ha.cf file** is the HeartBeat main configuration file consisting of a configuration parameters list. A sample follows:

```
bcast eth1
keepalive 2
warntime 10
deadtime 30
initdead 120
udpport 694
nice_failback on
node EY-FTP1          ##  This is your server 1
node EY-FTP2          ##  This is your server 2
ping 192.168.38.1     ##  This is a pingable address
```

The “node” definitions allow HeartBeat to identify the two servers that form the High Availability cluster. The last entry is how HeartBeat confirms its network connectivity. If the “ping” fails to operate correctly, HeartBeat assumes that although it is functioning correctly, it has lost connectivity to the network. In this case, HeartBeat proceeds to interchange to the other node/server.

Configuration File: haresources

This file contains information about the resources to be enabled or disabled by HeartBeat when a node mode change is detected. A node change might be a change from slave to master or from master to slave. It is mandatory that this file be identical on both servers.

Here is a sample:

```
EY-FTP1 192.168.38.130 mv_ipteld mv_watchdogd
```

The first entry: “EY-FTP1”defines the default master server. This definition allows HeartBeat to stop and start gracefully without causing potential conflicts or ambiguous modes.

The second entry: “192.168.38.130” is the “shared” IP address. In this case, “192.168.38.130” is not configured to be assigned to either server, but is dynamically assigned by the HeartBeat application.

The third entry is the name of a script file to run when the state of a server changes. This entry is executed with the “start” parameter on a change to master, and a “stop” on a change to slave. In this situation, the “**mv_ipteld**” script starts and stops the MV_IPTelD and MV_WatchDog daemons.

Configuration File: authkeys

This file controls how the two servers in the High Availability cluster control access to each other. The following file shows a simple mechanism.

```
auth 2

2 crc
```

Important:

If this file does not have suitable permissions, authentication fails. A typical recommendation is “600" which only allows “root” to read or write to the file.

Controlling HeartBeat

The heartbeat Daemon can be controlled exactly like any other Linux daemon. Use the following syntax to control the Daemon from the command line:

```
/etc/init.d/heartbeat [status|start|stop]
```

By default, HeartBeat logs to “**/var/log/ha-log**”. Use this log to check for errors. To simulate interchange between the two servers, stop the primary server that uses this command:

```
/etc/init.d/heartbeat stop
```

The key point here is that since the IP address in the file must be the same as the far end of the link, this file cannot be the same on both ends. For example, Server A needs to have the IP address of Server B and vice-versa.

HeartBeat gives priority to the primary server and will activate it as soon as possible. Just restart heartbeat to revert to normal operation.

```
/etc/init.d/heartbeat start
```

Important:

To control the cluster, use only the HeartBeat controls described here. Do not manually start or stop the Avaya IP Telephone File Server Application and /or MV_WatchDog daemons.

Installing and Configuring RSYNC

RSYNC is a client/server application for synchronization of files across two servers. Standard Red Hat installation usually installs RSYNC, but RSYNC is supplied as an RPM if it is not already installed.

Enabling the RSYNC Server

By default, the RSYNC server is configured for access by the XINETD Daemon, but the default configuration is to have RSYNC disabled. To change this default, edit the file "**/etc/xinetd.d/rsync**". The first line indicates that "disabled" is set to "yes". Change this value to "no" then restart the XINETD service to activate the change.

RSYNCD.CONF

The file "**/etc/rsyncd.conf**" defines symbolic names for RSYNC and controls logging. A sample file follows:

```
log file = /var/log/rsyncd.log
[MV_IPTel_data]
   path = /opt/ecs/mvuser/MV_IPTel/data/FTPdata
   comment = MV FTP Data dir
   list = yes
   read only = no
[MV_IPTel_TFTPdata]
   path = /opt/ecs/mvuser/MV_IPTel/data/TFTPdata
   comment = MV TFTP Data dir
   list = yes
   read only = no
```

Optional:

```
[MV_IPTel_HTTPdata]
   path = /opt/ecs/mvuser/MV_IPTel/data/HTTPdata
   comment = MV HTTP Data dir
   list = yes
   read only = no
```

This file defines the two/three directories to be synchronized between the two servers. Note that this only defines the operation of RSYNC and what symbolic names are available, it does not perform any actual synchronization.

RSYNCHOURLY.SH

RSYNC must be executed with suitable parameters to perform synchronization. The sample shell script that follows directs a single synchronization process. CRON can also use the script on an hourly basis to ensure that both servers are synchronized regularly.

```
rsync -qruW 10.0.0.1::MV_IPTel_data/ /opt/ecs/mvuser/MV_IPTel/data/
  FTPdata

rsync -qruW 10.0.0.1::MV_IPTel_TFTPdata/ /opt/ecs/mvuser/MV_IPTel/
  data/TFTPdata
```

Optional:

```
rsync -qruW 10.0.0.1::MV_IPTel_HTTPdata/ /opt/ecs/mvuser/MV_IPTel/
  data/HTTPdata
```

Copy this file to "**/etc/cron.hourly**". Restart the CRON daemon to regularly execute the job. This file is NOT the same for both servers. The IP address at the start of the parameters is the source of files on the remote server, and so will be the "other" server for both servers. To clarify this, server A must have server B's IP address and vice-versa. The remaining arguments define the RSYNC symbolic name on the "other" machine and the local file structure to be used as a target for RSYNC.

Network Time Protocol

Although Network Time Protocol (NTP) is not an essential component of the High Availability server cluster, NTP allows the two servers to have identical times. Having identical times helps when analyzing log files and prevents problems with the RSYNC process.

The Network Time Protocol Daemon (NTPD) is installed as standard for Linux, but is not usually enabled. After suitable configuration, enable and configure the NTPD service to start automatically following a Linux restart.

NTP.CONF

NTP.CONF is the configuration file for the Network Time Protocol Daemon. This file has many elements however, only a few defaults need to change as underlined in the following sample file.

```
# Prohibit general access to this service.
# Enable NTP on this server
restrict default nomodify notrap noquery
# Permit all access over the loopback interface.  This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
restrict 127.0.0.1
# -- CLIENT NETWORK -------
# Permit systems on this network to synchronize with this
# time service.  Do not permit those systems to modify the
# configuration of this service.  Also, do not use those
# systems as peers for synchronization.
# restrict 192.168.1.0 mask 255.255.255.0 notrust nomodify notrap
# --- OUR TIMESERVERS -----
# or remove the default restrict line
# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.
# restrict mytrustedtimeserverip mask 255.255.255.255 nomodify
  notrap noquery
# server mytrustedtimeserverip
server ntp.cis.strath.ac.uk     ## The Time server should local #
  --- NTP MULTICASTCLIENT ---
#multicastclient# listen on default 224.0.1.1
# restrict 224.0.1.1 mask 255.255.255.255 notrust nomodify notrap
# restrict 192.168.1.0 mask 255.255.255.0 notrust nomodify notrap
# --- GENERAL CONFIGURATION ---
#
# Undisciplined Local Clock. This is a fake driver intended for
  backup
# and when no outside source of synchronized time is available. The
# default stratum is usually 3, but in this case we elect to use
  stratum
# 0. Since the server line does not have the prefer keyword, this
  driver
# is never used for synchronization, unless no other
# synchronization source is available. In case the local host is
# controlled by some external source, such as an external oscillator
  or
# another protocol, the prefer keyword would cause the local host to
```

```
# disregard all other synchronization sources, unless the kernel
# modifications are in use and declare an unsynchronized condition.
#
#server127.127.1.0# local clock
#fudge127.127.1.0 stratum 10
#
# Drift file.  Put this in a directory which the daemon can write
  to.
# No symbolic links allowed, either, since the daemon updates the
  file
# by creating a temporary in the same directory and then
  rename()'ing
# it to the file.
#
driftfile /etc/ntp/drift
broadcastdelay0.008
#
# Authentication delay.  If you use, or plan to use someday, the
# authentication facility you should make the programs in the
  auth_stuff
# directory and figure out what this number should be on your
  machine.
#
authenticate yes
#
# Keys file.  If you want to check your server at run time, make a
# keys file (mode 600 for sure) and define the key number to be
# used for making requests.
#
# PLEASE DO NOT USE THE DEFAULT VALUES HERE. Pick your own, or
  remote
# systems might be able to reset your clock at will. Note also that
# ntpd is started with a -A flag, disabling authentication, that
# will have to be removed as well.
#
keys/etc/ntp/keys
```

Use this command to validate the NTPD operation:

```
Ntpq –c peers
```

Index

Symbols

A

B

C

D

F

H

I

L

M

O

P

R

S

T

U

W