FTP Settings

When first getting started running an FTP server, most people find that Rumpus works very well in it's default configuration. Over time, however, you may decide to alter certain aspects of your server's operation by adjusting the server-wide options that are available.

Basic FTP Settings

Address To Serve On

Mac servers can be configured with multiple IP addresses, and this setting allows you to choose one for FTP serving, if needed. By default, Rumpus will serve on any and all configured IP addresses, so if your server has only one address assigned, or if FTP services are needed for all configured addresses, this setting should be left at the default. However, if your server has multiple IP addresses and you want FTP services enabled on only one address, specify that address here. Connection attempts to all other configured addresses will then be ignored by Rumpus.

If you specify an address, but later wish to return Rumpus to the default of serving on all configured addresses, simply enter "all" as the address to serve on.

Port Number

The default port used for FTP services is 21, but you may change this if necessary. For example, if you want to run Rumpus on a nonstandard port as a security measure or so that anonymous FTP users cannot easily establish a connection.

However, you cannot use any port number you wish. The general practice is that port numbers 1 through 1024 are reserved and should not be used as alternatives to the default port. Instead, the accepted practice is to multiply the default port by 100, then add one. For example, you've probably seen the nonstandard port of 8001 used by some Web servers, even though the default port is 80. For FTP servers, try using 2101 for the first FTP server using a nonstandard port, then 2102, 2103, etc.

To access Rumpus using a nonstandard port, users will need to enter the domain name or IP number of the server followed by a space, followed by the new port number. Your users will need to know the nonstandard port number and how to configure an FTP client in order to access Rumpus configured with a nonstandard port, so use it wisely. In general, unless you have a specific need to use a nonstandard port, we strongly recommend that the default port of 21 be

used.

Maximum Simultaneous Connections

Rumpus can have a maximum of 32 simultaneous connections (or 256 using Rumpus Pro), which is defined as the total number of users connected to the server at any one time. Configure this setting to a number that your network is realistically capable of supporting. Also, if your FTP server is also providing other services such as Web and AppleShare, consider reducing the number of simultaneous connections to a level that will not interfere with your server's other duties.

FTP Root Folder

The "FTP Root Folder" is the default top-level folder available to users via your FTP server. Unless a drop folder is specified for a user, the FTP Root Folder is where clients will be placed when they first log in to Rumpus. Most FTP clients will immediately display the root folder's directory listing after login, and users will then be able to upload, download, and navigate in the root folder and it's sub-folders.

Timeouts

In order to keep available connections from being indefinitely tied up by inactive or incorrectly disconnected clients, Rumpus allows you to specify several timeout values. These values indicate the time, in seconds, Rumpus should allow various network actions to take place before determining that a problem has occurred and disconnecting the user.

For the "User Inactivity" timeout value, specify any setting that makes sense for the maximum amount of time for a client to be inactive when attached to your server. For an extremely busy site where users should be uploading or downloading one or a few files, the time-out can be set to a relatively low value. If, on the other hand, there are a relatively small number of users who are accessing the FTP server for longer periods of time, it makes more sense to set a relatively long time-out period so that users do not have to frequently reconnect.

Connection time-out values can be adjusted to suit your networking environment. Time-outs can be set separately for the time permitted to open a new connection, send a single packet of data, receive a single packet of data, and close the connection. Connection open and close operations must be performed completely within the defined time period, or Rumpus will terminate the connection. For send and receive operations, the time-out specifies the maximum amount of time allowed for a single TCP/IP packet to be sent or received. It is important to note that file transfers, especially large file transfers, may take substantially longer than the time-out setting indicates. Transfers will

continue without timing out as long as individual packets are transferred at least once within the time-out period specified. The default value for each of the time-out settings is 60 seconds.

Security Options

Hack Attempt Recognition

If you would like Rumpus to track login attempts and disable computers that appear to be trying to hack their way into your server, enable the "Auto-Detect Hack Attempts" option. Rumpus can be made to be more, or less, aggressive about detecting problems by increasing or decreasing the number and frequency of the failed connection attempts that must be made to cause the address to be added to the block list. The "Block After" field sets the number of successive connection failures that must be reached to cause an address to be added to the denied list, while the "Within" field allows you to specify the time period in which the failed attempts must occur.

Note that when a hack attempt is automatically detected, the address is added to the block list exactly as if you had added it yourself. So, if Rumpus makes a mistake and disables a computer that legitimately needs FTP access to your server, remove the address from the blocked list using the "Blocked Clients" window.

Disable 3rd Party PORT Connections

One seldom used but sometimes handy feature of FTP is the ability to transfer files between 2 FTP servers directly, without the FTP client having to download and resend the file data. Unfortunately, this capability can potentially be used by unauthorized people to transfer large amounts of data between servers, even when their own bandwidth is limited. This is a concern particularly for administrators of FTP servers that allow anonymous users to upload and store data on the server.

If you would like to disallow server to server transfers, check the "Disable 3rd Party PORT Connections" option. When enabled, this security measure causes Rumpus to reject active mode data connections to any IP address other than the client making the request. In other words, when a client makes a PORT request to establish a data connection, Rumpus will not establish the connection to a 3rd party server, or any computer other than the controlling client.

Disallow Direct File Aliases On Server

Aliased folders are always resolved, secured, and served as you would expect. However, an alias that leads directly to a file (not to a folder or volume) will not be served when this option is selected. Serving files aliased directly can represent a

security risk because it is not always clear which folder path to the file should be used in considering the user's access permissions. In general, it is best to avoid this confusion entirely and simply make a full copy of any file needed directly to the FTP folder hierarchy.

Disable Uploaded Alias Files

One way in which hackers might attempt to compromise your server is by using special features of the operating system in combination with different server applications to create security holes. While unlikely, it is theoretically possible for Mac OS Finder Aliases to be used in such a scheme, because aliases can be made that might provide access to otherwise inaccessible folders on your hard disk.

To eliminate this possibility, Rumpus provides the option of disabling aliases that are uploaded through Rumpus FTP services. When MacBinary files are received, Rumpus checks the file to determine if it is a Finder alias. (Normal text and raw binary files cannot be aliases, so MacBinary is the only way Rumpus supports sending an alias in the first place.) If the uploaded file is an alias, Rumpus simply deactivates it. The file is still saved on the hard drive, but it will not function as an alias to any file or folder on the server.

Disable User Accounts After Several Failed Login Attempts

When checked, this option will disable user accounts after a succession of 4 failed login attempts. When a user attempts to log in to the FTP server with a valid username but an incorrect password, they are, of course, rejected. If such a login failure occurs 4 times in succession, the user account will be disabled. To re-enable the account, the server administrator must re-check the "Permit Login" security setting for the disabled account using either the Web-based administration or Mac OS interface. Note that if fewer than 4 login failures occur, and then the user successfully logs in, the "failed counter" is reset to 0.

Send Empty Listings Instead Of "Permission Denied"

Sometimes, users may request a directory listing in a folder when their user account does not permit such lists. This is most common when implementing drop boxes, where users are able to upload files to your server, but unable to download, delete, or see a listing of files already in the folder.

When a user is not allowed to see the contents of a folder, Rumpus will normally send an error response of Not Authorized to the client. Unfortunately, many clients see this error and misinterpret it, logging the user out of the server and effectively disabling the drop box.

When the Empty Dir Listings option is checked, Rumpus sends an empty

directory listing to the client instead of an error code. In this way, the contents of the directory are still hidden from users, but no error code is generated. The FTP client will then permit users to upload files as needed. If you are implementing a drop box, or simply want to send blank directory listings instead of returning a Not Authorized error response, be sure to check this option.

Make Missing Folders

This option tells Rumpus to create folders when file uploads specify folders that don't already exist. This resolves problems with some poorly behaved FTP clients, which may try to upload entire directories of files without first creating the directory into which the files are to be placed. Enabling this option is not usually a good idea, as it may result in simple upload errors that lead to files saved in incorrect and unexpected folders. However, when it is essential that misbehaved clients be supported, this option may be necessary.

Log Files

Logging connections and attempted connections to your FTP server is important for several reasons. First, you can determine what files are being accessed on your server the most, as well as when and by whom. Knowing user habits can help you plan to provide better services. Also, by examining the contents of your log files you can identify abuse of your system, such as illegal access attempts, improper deletion of files, and files that shouldn't be uploaded to your FTP server. The logging tab allows you to specify the folder where log files should be stored, and to determine which log files you would like to have maintained.

Log Folder

Choose the folder where all Rumpus log files should be saved. You can set the folder by entering the folder path manually, or click the Choose Folder button and select the folder using the standard Mac folder selection box. The Open Folder button next to the field can be used to open the selected log folder in the Finder.

Maintain User Activity Log

When checked, a log file is maintained that records each user action. Information recorded includes IP address, the time and date of the connection, and the command issued by the user's FTP client, along with the result of each command.

Maintain Anonymous Password Log

When checked, a file is created for storing the passwords entered by anonymous users. People generally enter their e-mail address as their anonymous FTP

password, so this log can be useful for the Rumpus administrator for debugging and statistics.

Maintain Failed Access Log

Rumpus can maintain a log detailing any failed login attempt. This log can be used to warn you about unauthorized users attempting to hack into your FTP server. It will also warn you about server problems or potential errors. For example, the log will show you if users are being rejected due to the maximum number of simultaneous users.

Record Session Transcripts

When enabled, the Rumpus server will record each action requested by each user during a session, along with the response from the server. Transcripts can then be viewed using the Rumpus control application "Active Users" window or saved to the log files folder on the hard drive. Transcripts are different for FTP and WFM users, since the protocols differ in how client requests are made. For FTP users, the transcipt will include each FTP command issued by the client. For WFM users, a single line for each HTTP request will be recorded, indicating the file requested and how the request was handled.

Since session transcripts record every action taken by every user over the entire course of their session, recording these transcripts will increase Rumpus' memory and CPU usage. On lightly used servers, the extra memory and CPU usage will be minimal, while on very busy servers, the impact will be more severe. If you have a need to maintain transcripts for logging purposes or to properly manage the server, then be sure to enable this feature of Rumpus. However, if there is no compelling need to review transcipts on a regular basis, consider turning this feature off during normal use to optimize services. Of course the transcript record can be turned on and off for occassional monitoring as needed.

Include Directory List Contents

If you would like to have the full contents of FTP directory listings included in the session transcript, enable this option. Directory listings can quickly increase the size of session transcripts, so this option is provided to allow you to tune the amount of information recorded in the transcripts.

Save Session Transcripts

When session transcripts are being maintained, they can be saved to disk if needed. When the "Save Session Transcripts" option is enabled, transcripts will be saved to a file in the log files folder, named with the user account name followed by ".tscpt". Each new transcript saved will be added to the end of the

user's transcript file, if it already exists.

Automatic Log Rolling

Rumpus provides the automated capability of rolling the log files it produces. Log rolling refers to the practice of periodically moving log files to another location with a unique name, then continuing to save log entries to a new file. This prevents logs from growing to excessive lengths, and, when done by date, produces a series of log files each from a particular time frame.

Rumpus can roll logs on a daily, weekly, or monthly basis, and the old log files can be moved to any folder you choose. When logs are rolled daily, they are moved to the archive folder at midnight each day. Weekly rolling occurs at midnight each week on Saturday night. Monthly rolling archives log files at midnight on the last day of each month. Note that Rumpus must be running at the time the logs should be rolled for the files to actually be moved to the archive.

When Rumpus moves files, it appends the date the log file was rolled to the filename and stores the file in the specified log folder. You can then review the logs using a text editor, delete them, compress and archive them, or simply leave them in the log folder for permanent archival.

Messages

One of the legacies of command-line FTP clients is the use of a startup or login message. These messages were used to advise the user of timely or important information, such as acceptable use policy, server downtimes, how to request help, etc. Login messages aren't used as much these days because FTP clients that use a graphical interface sometimes do not display such messages, even though the information is sent by the server. Server messages are sometimes displayed in a separate window in GUI clients such as Fetch or Interarchy; others display them in the area where FTP commands are displayed, as many Windows clients do.

Welcome Message

Enter a text message to be sent when users initially connect and log in to the server.

Goodbye Message

Enter a text message to be sent when users log off from the server.

Display Folder Messages

If the "Display Folder Messages From" checkbox is selected, Rumpus will send the contents of the file with the specified name when a user accesses a particular

folder, including the root folder. Message files should be plain text with line breaks and are completely optional. Rumpus does not require a message file in every folder if this option is on, and will simply not send a message when the user moves into a folder that does not have a message file.

Hide Message Files In Directory Listings

If you use folder messages, but prefer that your clients not see the message files in their directory listings, check the "Hide Message Files" option. When this option is checked, directory listings sent to the client will not include a line for the folder's message file.

Encoding Options

The Encoding tab allows you to customize how files are encoded, saved, and displayed to clients for your particular environment. Note that MacBinary encoding is always available for clients that support and ask for it (and most Mac FTP clients are smart enough to do just that).

Text File Type, Text Creator Code

Mac OS files are labeled with "Type" and "Creator" codes, which enable the operating system to determine which applications are suitable for processing which files. When files are uploaded to your FTP server in plain text mode, Rumpus can save the file with appropriate type and creator codes so that the file can be opened and handled by any application you choose.

In the case of text files, it is unlikely that you will need to change the default Text Type, as "TEXT" is the standard type code for plain Mac OS text files. The Text Creator, however, can be set so that double-clicking on the file will open it in the application of your choice. For example, to have plain text files opened using SimpleText, set the Text Creator to "ttxt". To use BBEdit as the standard application for opening text files, set the code to "R*ch". It is important to note that both type and creator codes are case sensitive. Be sure to enter them exactly, including matching case.

Note that the type and creator codes specified here are used only as default values. When Rumpus can determine a specific type and creator code for a file, based on the filename suffix and a matching entry in the Suffix Mapping list, that code will be used. When the filename does not match an entry in the Suffix Mapping list, the defaults specified here will be used instead.

Binary File Type, Binary Creator Code

FTP clients can also upload raw binary files to your server, in addition to text format, as described above. Just as with text files, Rumpus allows you to select

the type and creator codes for these binary files. Because the contents of binary files vary, sensible default values for Binary Type and Creator are impossible. However, these values can be set for your site if most of the binary uploads are of a common file type. For example, if GIF images are frequently uploaded as raw binary files, you might choose to set the Binary Type to "GIFf" and the Binary Creator to "GKON". This would cause raw binary files to be saved as Graphic Converter GIF image files.

Note that the type and creator codes specified here are used only as default values. When Rumpus can determine a specific type and creator code for a file, based on the filename suffix and a matching entry in the Suffix Mapping list, that code will be used. When the filename does not match an entry in the Suffix Mapping list, the defaults specified here will be used instead.

Important Note! Most Mac OS clients will send files using the MacBinary file format. Files encoded with MacBinary are transferred with their type and creator codes intact, and Rumpus will save the files using the correct, original codes. Both the text and binary type and creator code settings in Rumpus will apply only when the codes cannot be determined using the file encoding, such as when files are uploaded from a non-Mac OS FTP client.

Show Actual Filenames In Lists

This option allows you to display or suppress the actual filenames of files in directory listings ("lists"). In most cases, you should leave this option on, so that filenames are displayed correctly when clients request directory listings. However, by un-checking this option, you can suppress the real filenames from being shown. This feature can be useful when you want to display filenames with either the ".bin" and/or ".hqx" extensions only, as described below.

Show ".bin" Extension In Lists

As discussed below, Rumpus can automatically serve a file in MacBinary format when asked by simply appending the ".bin" extension to the filename of the file being requested. Many clients, however, will not be able to use this functionality simply because they do not know it exists (or because their graphical FTP client will not allow it). For this reason, Rumpus can automatically duplicate each filename in a directory listing, appending a ".bin" suffix to the extra filename entry. This will make it appear that each file in a directory actually includes an extension of ".bin", duplicating each file listing when "Show Actual Filenames In Lists" is also enabled.

Show ".hqx" Extension In Lists

In the same way that Rumpus can append ".bin" extensions to filenames in directory listings (see "Show '.bin' Extension In Lists" above), the extension ".hqx"

can be automatically added to each filename in a list. While the modified filename will not actually exist as a file on the server, Rumpus will serve the file as a binhexed file correctly as expected.

Allow Automatic MacBinary Encoding

MacBinary is a long-standing Mac OS protocol for maintaining Mac-specific portions of files when they are transferred over non-Mac specific connections. In other words, MacBinary allows Mac files to be transferred over FTP as Mac files, with their resource forks and Finder information intact. If your server will be supporting Mac OS clients, this is a very important capability.

MacBinary transfers are always available to Mac-savvy FTP client applications. However, if Mac OS users need to ensure that downloaded files are encoded in MacBinary, even when their FTP client doesn't specifically support it, they can simply add an extension of ".bin" to their file request, and Rumpus will send the file with the proper encoding. These files can then be opened with any one of several utilities, including Stuffit Expander, and saved permanently, complete with their resource fork and Finder information.

Allow Automatic BinHex Encoding

Automatic BinHex Encoding can be enabled so that users can add a suffix of ".hqx" to any filename request and Rumpus will automatically encode the file into BinHex format as it is returned. For example, to receive a file named ReadMe (which resides in the FTP folder) in BinHex format, the client can simply request the file ReadMe.hqx and Rumpus will BinHex the file as it is served.

BinHex is a popular format that puts the entire Macintosh file, including both data and resource forks, into a single, 7-bit data stream. While virtually all modern FTP clients support 8-bit binary transfer mode, BinHex can be used when only 7-bit ASCII mode is available (due to client or network limitations). The format can be decoded by many popular compression engines, including StuffIt Expander.

Save ASCII Uploads With Unix Line Endings

This option causes files uploaded in ASCII mode to be saved with Unix-style line endings (linefeeds). When the option is unchecked, files will be saved with classic Mac OS line endings (carriage returns).

The option pertains only to ASCII mode transfers. Binary mode transfers, including MacBinary, binhex, and plain binary file uploads are not affected. Text files uploaded in binary mode will be saved with the same line ending character(s) as the original file sent from the client.

Advanced Options

The Advanced Options tab provides control over several settings which, if set improperly, can cause serious problems for you and your FTP users. Care must be taken when changing these settings, though Rumpus does provide the ability to easily reset them to the "factory" defaults by clicking the "reset" button next to each field.

Passive Mode Connect Address

When attempting to use Rumpus to provide FTP service on computers without a dedicated IP address, or on servers that are behind a router that provides port-specific address translation, this field must be set correctly.

In the case where your LAN is configured with private addresses (non-routable addresses that can't be accessed from outside your local network), many routers can be configured to forward FTP traffic to your Rumpus server. In this case, enter the TCP/IP address of your router as the "Passive Mode Connect Address". When you do, Rumpus will direct FTP clients to the router for file transfers, and the router will in turn forward the incoming connections to Rumpus. For full details on FTP passive mode connections and the special networking needs of FTP servers, see the "FTP Overview" and "Port Forwarding" articles in the "Helpful Info" folder of the Rumpus package.

Use LAN Address For Passive Connections From Local Clients

In cases where a Passive Mode Connect Address is used to tell clients to connect to an external address to make passive mode connections, clients on your own LAN may be better served by accessing the server using the local address. When this option is enabled, external clients will be told to make passive connections to the external address, while LAN clients will be told to connect to the assigned local address of the server. This can improve local transfer speeds and conserve bandwidth, depending on your network configuration, and will usually simplify data routing on the network.

Reported FTP Server Name

Almost all graphical FTP clients use the "SYST" command to ask for information about the server software and operating system used by the server. This information is used by the client to optimize directory listings, file transfers, and other activities. The most notable example of this is when Mac OS clients recognize that the server is also running Mac OS, and then enable MacBinary mode file transfers.

The traditional response Mac FTP servers give to the "SYST" command is "MACOS Peter's Server". This response causes virtually every Macintosh FTP

client to enable MacBinary transfers, thereby allowing Mac files to remain Mac files after uploading or downloading. ("MACOS Peter's Server" was the name of the first MacBinary-capable FTP server developed years ago. Maxum did not choose the name, but defaults to it so that the server will be recognized as supporting the MacBinary format.)

If necessary, the name reported by Rumpus can be changed. Changing the name may result in inconsistent client behavior, since Mac clients may no longer be able to recognize that your FTP server is a Macintosh. At the very least, any reported name you decide to specify should begin "MacOS".

Passive Mode Port Range

The FTP protocol uses multiple connections to transfer files and directory listings. The difference between “Control” and “Data” connections, and between “Passive” and “Active” mode data connections can be found elsewhere in the Rumpus documentation. Passive mode data connections take place on ports other than the defined FTP control and active mode data ports of 20 and 21, and should always take place on generic port assignments above 1024. In addition, a range of ports must be specified, which allows the server to keep track of multiple simultaneous users by assigning each it’s own data port.

For various reasons, it might sometimes be useful to assign the port numbers yourself, usually to support network setups that restrict incoming connections to specific port ranges. To specify the port range, enter the lower bound of the range in the field provided. The upper bound is computed based on the number of simultaneous connections Rumpus is configured to support, and is displayed for your information.

Reserve 1 Connection For Administrator

This option allows you to save 1 connection for a specified user account (presumably the administrator). Check the "Reserve 1 Connection" checkbox, enter the name of any user account, and when the number of active FTP users reaches one less than the specified number of simultaneous connections, Rumpus will not allow additional logins. The defined user, however, will be allowed to connect to the maximum number of connections allowed.

Directory Listing Format For Files/Folders

Surprisingly, the FTP protocol does not specify a format for directory listings. In fact, the specification states that directory listings are human-readable only, will vary in format based on the operating system of the server, and should not be parsed by client applications. Unfortunately, graphical clients have no choice but to parse directory listings in order to function, making the format a server uses to send listings crucial to it's compatibility with the wide variety of clients available.

We have made every effort to format Rumpus directory listings such that they are parsable with virtually every significant FTP client. However, since there is no defined standard, and since the Mac OS differs from many other systems in several important ways (most notably the inclusion of Finder info and the existence of resource forks), occasional client incompatibilities may arise. These problems usually occur with special-purpose utilities and devices, not mainstream FTP clients, and so are actually quite rare. However, when an incompatibility does come up, Rumpus provides control over the directory listings so that knowledgeable server administrators can adjust the format as needed.

Making changes to the directory listing format is not a simple matter, and the ability to do so is provided only for highly knowledgable server administrators with very specific needs. If you are at all unsure about editing these options, please contact Maxum technical support for assistance.

The format of each line in a directory listing served by Rumpus is based on the strings configured here. Separate strings are specified for lines listing files and folders, since they differ in a number of important ways; however, they are processed by Rumpus exactly the same. Format strings are made up of text characters, spaces, and 4 digit codes (surrounded by angle brackets) that tell Rumpus where to put various bits of file information. Any insertion code can also include a size, forcing Rumpus to pad the text inserted to the specified length, allowing formatted columns to be maintained. For example, the command <size:8> will insert the file's size, padded with spaces so that it is always 8 characters long.

The available insertion codes are:

mode Inserts the file's permission (mode) bits, in traditional Unix fashion.

rssz Inserts the file's resource fork size.

dtsz Inserts the file's data fork size.

size Inserts the file's size, including both data and resource forks.

date Inserts the file's last modified date, in standard format.

name Inserts the name of the file.

objs Inserts the number of objects (files and folders) contained in a folder.