NAME

rsyncd.conf − configuration file for rsync in daemon mode

SYNOPSIS

rsyncd.conf

DESCRIPTION

The rsyncd.conf file is the runtime configuration file for rsync when run as an rsync daemon.

The rsyncd.conf file controls authentication, access, logging and available modules.

FILE FORMAT

The file consists of modules and parameters. A module begins with the name of the module in square brackets and continues until the next module begins. Modules contain parameters of the form 'name = value'.

The file is line-based -- that is, each newline-terminated line represents either a comment, a module name or a parameter.

Only the first equals sign in a parameter is significant. Whitespace before or after the first equals sign is discarded. Leading, trailing and internal whitespace in module and parameter names is irrelevant. Leading and trailing whitespace in a parameter value is discarded. Internal whitespace within a parameter value is retained verbatim.

Any line beginning with a hash (#) is ignored, as are lines containing only whitespace.

Any line ending in a \ is "continued" on the next line in the customary UNIX fashion.

The values following the equals sign in parameters are all either a string (no quotes needed) or a boolean, which may be given as yes/no, 0/1 or true/false. Case is not significant in boolean values, but is preserved in string values.

LAUNCHING THE RSYNC DAEMON

The rsync daemon is launched by specifying the **--daemon** option to rsync.

The daemon must run with root privileges if you wish to use chroot, to bind to a port numbered under 1024 (as is the default 873), or to set file ownership. Otherwise, it must just have permission to read and write the appropriate data, log, and lock files.

You can launch it either via inetd, as a stand-alone daemon, or from an rsync client via a remote shell. If run as a stand-alone daemon then just run the command "**rsync --daemon**" from a suitable startup script.

When run via inetd you should add a line like this to /etc/services:

```
rsync           873/tcp
```

and a single line something like this to /etc/inetd.conf:

```
rsync   stream  tcp     nowait  root    /usr/bin/rsync rsyncd --daemon
```

Replace "/usr/bin/rsync" with the path to where you have rsync installed on your system. You will then need to send inetd a HUP signal to tell it to reread its config file.

Note that you should **not** send the rsync daemon a HUP signal to force it to reread the `rsyncd.conf` file. The file is re-read on each client connection.

GLOBAL OPTIONS

The first parameters in the file (before a [module] header) are the global parameters.

You may also include any module parameters in the global part of the config file in which case the supplied value will override the default for that parameter.

motd file
: The "motd file" option allows you to specify a "message of the day" to display to clients on each connect. This usually contains site information and any legal notices. The default is no motd file.

log file The "log file" option tells the rsync daemon to log messages to that file rather than using syslog. This is particularly useful on systems (such as AIX) where syslog() doesn't work for chrooted programs. If the daemon fails to open to specified file, it will fall back to using syslog and output an error about the failure. (Note that a failure to open the specified log file used to be a fatal error.)

pid file The "pid file" option tells the rsync daemon to write its process ID to that file.

syslog facility
: The "syslog facility" option allows you to specify the syslog facility name to use when logging messages from the rsync daemon. You may use any standard syslog facility name which is defined on your system. Common names are auth, authpriv, cron, daemon, ftp, kern, lpr, mail, news, security, syslog, user, uucp, local0, local1, local2, local3, local4, local5, local6 and local7. The default is daemon.

port You can override the default port the daemon will listen on by specifying this value (defaults to 873). This is ignored if the daemon is being run by inetd, and is superseded by the **--port** command-line option.

address
: You can override the default IP address the daemon will listen on by specifying this value. This is ignored if the daemon is being run by inetd, and is superseded by the **--address** command-line option.

socket options
: This option can provide endless fun for people who like to tune their systems to the utmost degree. You can set all sorts of socket options which may make transfers faster (or slower!). Read the man page for the setsockopt() system call for details on some of the options you may be able to set. By default no special socket options are set.

MODULE OPTIONS

After the global options you should define a number of modules, each module exports a directory tree as a symbolic name. Modules are exported by specifying a module name in square brackets [module] followed by the options for that module.

comment
: The "comment" option specifies a description string that is displayed next to the module name when clients obtain a list of available modules. The default is no comment.

path The "path" option specifies the directory in the daemon's filesystem to make available in this module. You must specify this option for each module in `rsyncd.conf`.

use chroot
: If "use chroot" is true, the rsync daemon will chroot to the "path" before starting the file transfer with the client. This has the advantage of extra protection against possible implementation security holes, but it has the disadvantages of requiring super-user privileges, of not being able to follow symbolic links that are either absolute or outside of the new root path, and of complicating the

preservation of usernames and groups (see below). When "use chroot" is false, for security reasons, symlinks may only be relative paths pointing to other files within the root path, and leading slashes are removed from most absolute paths (options such as **--backup-dir**, **--compare-dest**, etc. interpret an absolute path as rooted in the module's "path" dir, just as if chroot was specified). The default for "use chroot" is true.

In order to preserve usernames and groupnames, rsync needs to be able to use the standard library functions for looking up names and IDs (i.e. getpwuid(), getgrgid(), getpwname(), and getgrnam()). This means a process in the chroot namespace will need to have access to the resources used by these library functions (traditionally /etc/passwd and /etc/group). If these resources are not available, rsync will only be able to copy the IDs, just as if the **--numeric-ids** option had been specified.

Note that you are free to setup user/group information in the chroot area differently from your normal system. For example, you could abbreviate the list of users and groups. Also, you can protect this information from being downloaded/uploaded by adding an exclude rule to the rsync.conf file (e.g. "exclude = /etc/**"). Note that having the exclusion affect uploads is a relatively new feature in rsync, so make sure your daemon is at least 2.6.3 to effect this. Also note that it is safest to exclude a directory and all its contents combining the rule "/some/dir/" with the rule "/some/dir/**" just to be sure that rsync will not allow deeper access to some of the excluded files inside the directory (rsync tries to do this automatically, but you might as well specify both to be extra sure).

max connections

The "max connections" option allows you to specify the maximum number of simultaneous connections you will allow. Any clients connecting when the maximum has been reached will receive a message telling them to try later. The default is 0 which means no limit. See also the "lock file" option.

max verbosity

The "max verbosity" option allows you to control the maximum amount of verbose information that you'll allow the daemon to generate (since the information goes into the log file). The default is 1, which allows the client to request one level of verbosity.

lock file

The "lock file" option specifies the file to use to support the "max connections" option. The rsync daemon uses record locking on this file to ensure that the max connections limit is not exceeded for the modules sharing the lock file. The default is `/var/run/rsyncd.lock`.

read only

The "read only" option determines whether clients will be able to upload files or not. If "read only" is true then any attempted uploads will fail. If "read only" is false then uploads will be possible if file permissions on the daemon side allow them. The default is for all modules to be read only.

write only

The "write only" option determines whether clients will be able to download files or not. If "write only" is true then any attempted downloads will fail. If "write only" is false then downloads will be possible if file permissions on the daemon side allow them. The default is for this option to be disabled.

list The "list" option determines if this module should be listed when the client asks for a listing of available modules. By setting this to false you can create hidden modules. The default is for modules to be listable.

uid The "uid" option specifies the user name or user ID that file transfers to and from that module should take place as when the daemon was run as root. In combination with the "gid" option this determines what file permissions are available. The default is uid -2, which is normally the user

"nobody".

gid The "gid" option specifies the group name or group ID that file transfers to and from that module should take place as when the daemon was run as root. This complements the "uid" option. The default is gid -2, which is normally the group "nobody".

filter The "filter" option allows you to specify a space-separated list of filter rules that the daemon will not allow to be read or written. This is only superficially equivalent to the client specifying these patterns with the **--filter** option. Only one "filter" option may be specified, but it may contain as many rules as you like, including merge-file rules. Note that per-directory merge-file rules do not provide as much protection as global rules, but they can be used to make **--delete** work better when a client downloads the daemon's files (if the per-dir merge files are included in the transfer).

exclude

The "exclude" option allows you to specify a space-separated list of patterns that the daemon will not allow to be read or written. This is only superficially equivalent to the client specifying these patterns with the **--exclude** option. Only one "exclude" option may be specified, but you can use "-" and "+" before patterns to specify exclude/include.

Because this exclude list is not passed to the client it only applies on the daemon: that is, it excludes files received by a client when receiving from a daemon and files deleted on a daemon when sending to a daemon, but it doesn't exclude files from being deleted on a client when receiving from a daemon.

exclude from

The "exclude from" option specifies a filename on the daemon that contains exclude patterns, one per line. This is only superficially equivalent to the client specifying the **--exclude-from** option with an equivalent file. See the "exclude" option above.

include

The "include" option allows you to specify a space-separated list of patterns which rsync should not exclude. This is only superficially equivalent to the client specifying these patterns with the **--include** option because it applies only on the daemon. This is useful as it allows you to build up quite complex exclude/include rules. Only one "include" option may be specified, but you can use "+" and "-" before patterns to switch include/exclude. See the "exclude" option above.

include from

The "include from" option specifies a filename on the daemon that contains include patterns, one per line. This is only superficially equivalent to the client specifying the **--include-from** option with a equivalent file. See the "exclude" option above.

auth users

The "auth users" option specifies a comma and space-separated list of usernames that will be allowed to connect to this module. The usernames do not need to exist on the local system. The usernames may also contain shell wildcard characters. If "auth users" is set then the client will be challenged to supply a username and password to connect to the module. A challenge response authentication protocol is used for this exchange. The plain text usernames are passwords are stored in the file specified by the "secrets file" option. The default is for all users to be able to connect without a password (this is called "anonymous rsync").

See also the "CONNECTING TO AN RSYNC DAEMON OVER A REMOTE SHELL PROGRAM" section in rsync(1) for information on how handle an rsyncd.conf-level username that differs from the remote-shell-level username when using a remote shell to connect to an rsync daemon.

secrets file

The "secrets file" option specifies the name of a file that contains the username:password pairs used for authenticating this module. This file is only consulted if the "auth users" option is specified. The file is line based and contains username:password pairs separated by a single colon. Any line starting with a hash (#) is considered a comment and is skipped. The passwords can contain any characters but be warned that many operating systems limit the length of passwords that can be typed at the client end, so you may find that passwords longer than 8 characters don't work.

There is no default for the "secrets file" option, you must choose a name (such as `/etc/rsyncd.secrets`). The file must normally not be readable by "other"; see "strict modes".

strict modes

The "strict modes" option determines whether or not the permissions on the secrets file will be checked. If "strict modes" is true, then the secrets file must not be readable by any user ID other than the one that the rsync daemon is running under. If "strict modes" is false, the check is not performed. The default is true. This option was added to accommodate rsync running on the Windows operating system.

hosts allow

The "hosts allow" option allows you to specify a list of patterns that are matched against a connecting clients hostname and IP address. If none of the patterns match then the connection is rejected.

Each pattern can be in one of five forms:

- a dotted decimal IPv4 address of the form a.b.c.d, or an IPv6 address of the form a:b:c::d:e:f. In this case the incoming machine's IP address must match exactly.
- an address/mask in the form ipaddr/n where ipaddr is the IP address and n is the number of one bits in the netmask. All IP addresses which match the masked IP address will be allowed in.
- an address/mask in the form ipaddr/maskaddr where ipaddr is the IP address and maskaddr is the netmask in dotted decimal notation for IPv4, or similar for IPv6, e.g. ffff:ffff:ffff:ffff:: instead of /64. All IP addresses which match the masked IP address will be allowed in.
- a hostname. The hostname as determined by a reverse lookup will be matched (case insensitive) against the pattern. Only an exact match is allowed in.
- a hostname pattern using wildcards. These are matched using the same rules as normal unix filename matching. If the pattern matches then the client is allowed in.

Note IPv6 link-local addresses can have a scope in the address specification:

```
fe80::1%link1
fe80::%link1/64
fe80::%link1/ffff:ffff:ffff:ffff::
```

You can also combine "hosts allow" with a separate "hosts deny" option. If both options are specified then the "hosts allow" option s checked first and a match results in the client being able to connect. The "hosts deny" option is then checked and a match means that the host is rejected. If the host does not match either the "hosts allow" or the "hosts deny" patterns then it is allowed to connect.

The default is no "hosts allow" option, which means all hosts can connect.

hosts deny

The "hosts deny" option allows you to specify a list of patterns that are matched against a connecting clients hostname and IP address. If the pattern matches then the connection is rejected. See the

"hosts allow" option for more information.

The default is no "hosts deny" option, which means all hosts can connect.

ignore errors

The "ignore errors" option tells rsyncd to ignore I/O errors on the daemon when deciding whether to run the delete phase of the transfer. Normally rsync skips the **--delete** step if any I/O errors have occurred in order to prevent disastrous deletion due to a temporary resource shortage or other I/O error. In some cases this test is counter productive so you can use this option to turn off this behavior.

ignore nonreadable

This tells the rsync daemon to completely ignore files that are not readable by the user. This is useful for public archives that may have some non-readable files among the directories, and the sysadmin doesn't want those files to be seen at all.

transfer logging

The "transfer logging" option enables per-file logging of downloads and uploads in a format somewhat similar to that used by ftp daemons. The daemon always logs the transfer at the end, so if a transfer is aborted, no mention will be made in the log file.

If you want to customize the log lines, see the "log format" option.

log format

The "log format" option allows you to specify the format used for logging file transfers when transfer logging is enabled. The format is a text string containing embedded single-character escape sequences prefixed with a percent (%) character. An optional numeric field width may also be specified between the percent and the escape letter (e.g. "%-50n %8l %07p").

The default log format is "%o %h [%a] %m (%u) %f %l", and a "%t [%p] " is always prefixed when using the "log file" option. (A perl script that will summarize this default log format is included in the rsync source code distribution in the "support" subdirectory: rsyncstats.)

The single-character escapes that are understood are as follows:

- %h for the remote host name
- %a for the remote IP address
- %l for the length of the file in bytes
- %p for the process ID of this rsync session
- %o for the operation, which is "send", "recv", or "del." (the latter includes the trailing period)
- %f for the filename (long form on sender; no trailing "/")
- %n for the filename (short form; trailing "/" on dir)
- %L either the string " -> SYMLINK", or " => HARDLINK" or an empty string (where **SYMLINK** or **HARDLINK** is a filename)
- %P for the module path
- %m for the module name
- %t for the current date time
- %u for the authenticated username (or the null string)
- %b for the number of bytes actually transferred
- %c when sending files this gives the number of checksum bytes received for this file

o %i an itemized list of what is being updated

For a list of what the characters mean that are output by "%i", see the **--itemize-changes** option in the rsync manpage.

Note that some of the logged output changes when talking with older rsync versions. For instance, deleted files were only output as verbose messages prior to rsync 2.6.4.

timeout

The "timeout" option allows you to override the clients choice for I/O timeout for this module. Using this option you can ensure that rsync won't wait on a dead client forever. The timeout is specified in seconds. A value of zero means no timeout and is the default. A good choice for anonymous rsync daemons may be 600 (giving a 10 minute timeout).

refuse options

The "refuse options" option allows you to specify a space-separated list of rsync command line options that will be refused by your rsync daemon. You may specify the full option name, its one-letter abbreviation, or a wild-card string that matches multiple options. For example, this would refuse **--checksum** (**-c**) and all the various delete options:

```
refuse options = c delete
```

The reason the above refuses all delete options is that the options imply **--delete**, and implied options are refused just like explicit options. As an additional safety feature, the refusal of "delete" also refuses **remove-sent-files** when the daemon is the sender; if you want the latter without the former, instead refuse "delete-*" -- that refuses all the delete modes without affecting **--remove-sent-files**.

When an option is refused, the daemon prints an error message and exits. To prevent all compression, you can use "dont compress = *" (see below) instead of "refuse options = compress" to avoid returning an error to a client that requests compression.

dont compress

The "dont compress" option allows you to select filenames based on wildcard patterns that should not be compressed during transfer. Compression is expensive in terms of CPU usage so it is usually good to not try to compress files that won't compress well, such as already compressed files.

The "dont compress" option takes a space-separated list of case-insensitive wildcard patterns. Any source filename matching one of the patterns will not be compressed during transfer.

The default setting is `*.gz *.tgz *.zip *.z *.rpm *.deb *.iso *.bz2 *.tbz`

AUTHENTICATION STRENGTH

The authentication protocol used in rsync is a 128 bit MD4 based challenge response system. Although I believe that no one has ever demonstrated a brute-force break of this sort of system you should realize that this is not a "military strength" authentication system. It should be good enough for most purposes but if you want really top quality security then I recommend that you run rsync over ssh.

Also note that the rsync daemon protocol does not currently provide any encryption of the data that is transferred over the connection. Only authentication is provided. Use ssh as the transport if you want encryption.

Future versions of rsync may support SSL for better authentication and encryption, but that is still being investigated.

EXAMPLES

A simple rsyncd.conf file that allow anonymous rsync to a ftp area at `/home/ftp` would be:

[ftp]

```
    path = /home/ftp
    comment = ftp export area
```

A more sophisticated example would be:

```
uid = nobody
gid = nobody
use chroot = no
max connections = 4
syslog facility = local5
pid fi le = /var/run/rsyncd.pid

[ftp]
    path = /var/ftp/pub
    comment = whole ftp area (approx 6.1 GB)

[sambaftp]
    path = /var/ftp/pub/samba
    comment = Samba ftp area (approx 300 MB)

[rsyncftp]
    path = /var/ftp/pub/rsync
    comment = rsync ftp area (approx 6 MB)

[sambawww]
    path = /public_html/samba
    comment = Samba WWW pages (approx 240 MB)

[cvs]
    path = /data/cvs
    comment = CVS repository (requires authentication)
    auth users = tridge, susan
    secrets fi le = /etc/rsyncd.secrets
```

The /etc/rsyncd.secrets fi le would look something like this:

```
tridge:mypass
susan:herpass
```

FILES

/etc/rsyncd.conf or rsyncd.conf

SEE ALSO

rsync(1)

DIAGNOSTICS

BUGS

Please report bugs! The rsync bug tracking system is online at http://rsync.samba.org/

VERSION

This man page is current for version 2.6.6 of rsync.

CREDITS

rsync is distributed under the GNU public license. See the file COPYING for details.

The primary ftp site for rsync is ftp://rsync.samba.org/pub/rsync.

A WEB site is available at http://rsync.samba.org/

We would be delighted to hear from you if you like this program.

This program uses the zlib compression library written by Jean-loup Gailly and Mark Adler.

THANKS

Thanks to Warren Stanley for his original idea and patch for the rsync daemon. Thanks to Karsten Thygesen for his many suggestions and documentation!

AUTHOR

rsync was written by Andrew Tridgell and Paul Mackerras. Many people have later contributed to it.

Mailing lists for support and development are available at http://lists.samba.org