# Configuration Directive List

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

# Configuration Directive List

***Table of Contents***

*DirFakeMode −− Hide real file/directory permissions*
*DirFakeUser −− Hide real file/directory owner*
*DisplayConnect −− Sets connect banner file*
*DisplayFirstChdir −− FIXME FIXME*
*DisplayGoAway −− FIXME FIXME*
*DisplayLogin −− FIXME FIXME*
*DisplayQuit −− FIXME FIXME*
*DisplayReadme −− FIXME FIXME*
*ExtendedLog −− FIXME FIXME*
*FileRatioErrMsg −− FIXME FIXME*
*FooBarDirective −− FIXME FIXME*
*Global −− FIXME FIXME*
*Group −− FIXME FIXME*
*GroupOwner −− FIXME FIXME*
*GroupPassword −− FIXME FIXME*
*GroupRatio −− Ratio directive*
*HiddenStor −− Enables more safe file uploads*
*HideGroup −− FIXME FIXME*
*HideNoAccess −− Block the listing of directory entries to which the user has no access permissions*
*HideUser −− FIXME FIXME*
*HostRatio −− Ratio directive*
*IdentLookups −− Toggle ident lookups*
*IgnoreHidden −− Treat 'hidden' files as if they don't exist*
*Include −− Load additional configuration directives from a file*
*LDAPAuthBinds −− FIXME FIXME*
*LDAPDefaultAuthScheme −− Set the authentication scheme/hash that is used when no leading {hashname} is present.*
*LDAPDefaultGID −− Set the default GID to be assigned to users when no uidNumber attribute is found.*
*LDAPDefaultUID −− Set the default GID to be assigned to users when no uidNumber attribute is found.*
*LDAPDNInfo −− Set DN information to be used for initial bind*
*LDAPDoAuth −− Enable LDAP authentication*
*LDAPDoGIDLookups −− Enable LDAP lookups for user group membership and GIDs in directory listings*
*LDAPDoUIDLookups −− Enable LDAP lookups for UIDs in directory listings*
*LDAPForceDefaultGID −− Force all LDAP−authenticated users to use the same GID.*
*LDAPForceDefaultUID −− Force all LDAP−authenticated users to use the same UID.*
*LDAPHomedirOnDemand −− Enable the creation of user home directories on demand*
*LDAPHomedirOnDemandPrefix −− Enable the creation of user home directories on demand*
*LDAPHomedirOnDemandPrefixNoUsername −− FIXFIXFIX*
*LDAPHomedirOnDemandSuffix −− Specify an additional directory to be created inside a user's home directory on demand.*
*LDAPNegativeCache −− Enable negative caching for LDAP lookups*
*LDAPQueryTimeout −− Set a timeout for LDAP queries*
*LDAPSearchScope −− Specify the search scope used in LDAP queries*
*LDAPServer −− Specify the LDAP server to use for lookups*
*LDAPUseTLS −− Enable TLS/SSL connections to the LDAP server.*
*LeechRatioMsg −− Sets the 'over ratio' error message*
*Limit −− FIXME FIXME*

*SQLAuthTypes* -- *FIXME FIXME*
*SQLConnectInfo* -- *FIXME FIXME*
*SQLDefaultGID* -- *FIXME FIXME*
*SQLDefaultHomedir* -- *FIXFIXFIX*
*SQLDefaultUID* -- *FIXME FIXME*
*SQLDoAuth* -- *FIXME FIXME*
*SQLDoGroupAuth* -- *FIXME FIXME*
*SQLEmptyPasswords* -- *Allow zero length passwords (DEPRECATED)*
*SQLEncryptedPasswords* -- *Assume SQL passwords are encrypted (DEPRECATED)*
*SQLGidField* -- *FIXFIXFIX*
*SQLGroupGIDField* -- *FIXFIXFIX*
*SQLGroupInfo* -- *FIXFIXFIX*
*SQLGroupMembersField* -- *FIXME FIXME*
*SQLGroupnameField* -- *FIXME FIXME*
*SQLGroupTable* -- *FIXME FIXME*
*SQLGroupWhereClause* -- *FIXFIXFIX*
*SQLHomedir* -- *FIXFIXFIX*
*SQLHomedirField* -- *FIXFIXFIX*
*SQLHomedirOnDemand* -- *FIXME FIXME*
*SQLLog* -- *FIXFIXFIX*
*SQLLogDirs* -- *FIXFIXFIX*
*SQLLogHits* -- *FIXFIXFIX*
*SQLLogHosts* -- *FIXFIXFIX*
*SQLLoginCountField* -- *FIXFIXFIX*
*SQLLogStats* -- *FIXFIXFIX*
*SQLMinID* -- *FIXME FIXME*
*SQLMinUserGID* -- *FIXFIXFIX*
*SQLMinUserUID* -- *FIXFIXFIX*
*SQLNamedQuery* -- *FIXFIXFIX*
*SQLPasswordField* -- *FIXFIXFIX*
*SQLProcessGrEnt* -- *FIXFIXFIX*
*SQLProcessPwEnt* -- *FIXFIXFIX*
*SQLRatios* -- *FIXFIXFIX*
*SQLRatioStats* -- *FIXFIXFIX*
*SQLScrambledPasswords* -- *FIXME FIXME*
*SQLShellField* -- *FIXME FIXME*
*SQLShowInfo* -- *FIXFIXFIX*
*SQLSSLHashedPasswords* -- *FIXME FIXME*
*SQLUidField* -- *FIXFIXFIX*
*SQLUserInfo* -- *FIXFIXFIX*
*SQLUsernameField* -- *FIXFIXFIX*
*SQLUserTable* -- *FIXFIXFIX*
*SQLUserWhereClause* -- *FIXFIXFIX*
*SQLWhereClause* -- *FIXME FIXME*
*SyslogFacility* -- *Set the facility level used for logging*
*SyslogLevel* -- *Set the verbosity level of system logging*
*SystemLog* -- *Redirect syslogging to a file*
*TCPAccessFiles* -- *Sets the access files to use*
*TCPAccessSyslogLevels* -- *Sets the logging levels for mod_wrap*
*tcpBackLog* -- *Control the tcp backlog in standalone mode*
*TCPGroupAccessFiles* -- *Sets the access files to use*

*tcpNoDelay −− Control the use of TCP_NODELAY*
*tcpReceiveWindow −− Set the size of the tcp receive window*
*tcpSendWindow −− Set the size of the tcp send window*
*TCPServiceName −− Configures the name proftpd will use with mod_wrap*
*TCPUserAccessFiles −− Sets the access files to use*
*TimeoutIdle −− Sets the idle connection timeout*
*TimeoutLogin −− Sets the login timeout*
*TimeoutNoTransfer −− Sets the connection without transfer timeout*
*TimeoutStalled −− Sets the timeout on stalled downloads*
*TimesGMT −− Toggle time display between GMT and local*
*TransferLog −− Specify the path to the transfer log*
*Umask −− Set the default Umask*
*UseFtpUsers −− Block based on /etc/ftpusers*
*UseGlobbing −− Toggles use of glob() functionality*
*User −− Set the user the daemon will run as*
*UserAlias −− Alias a username to a system user*
*UserDirRoot −− Set the chroot directory to a subdirectory of the anonymous server*
*UseReverseDNS −− Toggle rDNS lookups*
*UserOwner −− Set the user ownership of new files / directories*
*UserPassword −− Creates a hardcoded username/password pair*
*UserRatio −− Ratio directive*
*VirtualHost −− Define a virtual ftp server*
*WtmpLog −− Toggle logging to wtmp*

*2. List of modules*

*mod_auth −− Authentication module*
*mod_code −− FIX ME FIX ME*
*mod_core −− Core module*
*mod_ldap −− LDAP authentication support*
*mod_log −− Logging support*
*mod_ls −− file listing functionality*
*mod_pam −− Pluggable authentication modules support*
*mod_quota −− Module to implement per−user quotas*
*mod_ratio −− FIX ME FIX ME*
*mod_readme −− "README" file support*
*mod_sample −− Example module*
*mod_site −− FIX ME FIX ME*
*mod_sql −− SQL support module*
*mod_unixpw −− UNIX style authentication methods*
*mod_wrap −− Interface to libwrap*
*mod_xfer −− FIX ME FIX ME*

*3. List of configuration contexts*

*server config −− server config*
*Global −− Global*
*VirtualHost −− VirtualHost*
*Anonymous −− Anonymous*
*Limit −− Limit*
*.ftpaccess −− .ftpaccess*

# Chapter 1. List of Directives

# AccessDenyMsg

## Name

AccessDenyMsg –– Customise the response on failed authentication

## Synopsis

**AccessDenyMsg** [ `"message"`]

`Default`
Dependent on login type
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.2.2 and later

## Description

Normally, a 530 response message is sent to an FTP client immediately after a failed authentication attempt, with a standard message indicating the the reason of failure. In the case of a wrong password, the reason is usually "Login incorrect." It is this message can be customized with the AccessDenyMsg directive. In the message argument, the magic cookie '%u' is replaced with the username specified by the client during login.

## See also

## Examples

AccessDenyMsg "Guest access denied for %u."

# AccessGrantMsg

## Name

AccessGrantMsg –– Customise the response on successful authentication

## Synopsis

**AccessGrantMsg** [ `"message"`]

`Default`
Dependent on login type
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0pl5 and later

## Description

Normally, a 230 response message is sent to an FTP client immediately after authentication, with a standard message indicating that the user has either logged in or that anonymous access has been granted. This message can be customized with the AccessGrantMsg directive. In the message argument, the magic cookie '%u' is replaced with the username specified by the client during login.

## See also

## Examples

AccessGrantMsg "Guest access granted for %u."

# Allow

## Name

Allow –– Access control directive

## Synopsis

**Allow** [ ["from"] "all"|"none"|host|network[,host|network[,...]]]

Default
: Allow from all

Context
: <Limit>

Module
: mod_core

Compatibility
: 0.99.0pl6 and later

## Description

The Allow directive is used inside a <Limit> context to explicitly specify which hosts and/or networks have access to the commands or operations being limited. Allow is typically used in conjunction with Order and Deny in order to create sophisticated (or perhaps not–so–sophisticated) access control rules. Allow takes an optional first argument; the keyword from. Using from is purely cosmetic. The remaining arguments are expected to be a list of hosts and networks which will be explicitly granted access. The magic keyword all can be used to indicate that all hosts will explicitly be granted access (analogous to the AllowAll directive, except with a lower priority). Additionally, the magic keyword none can be used to indicate that no hosts or networks will be explicitly granted access (although this does not prevent them from implicitly being granted access). If all or none is used, no other hosts or networks can be supplied. Host and network addresses can be specified by name or numeric address. For security reasons, it is recommended that all address information be supplied numerically. Relying solely on named addresses causes security to depend a great deal upon DNS servers which may themselves be vulnerable to attack or spoofing. Numeric addresses which specify an entire network should end in a trailing period (i.e. 10.0.0. for the entire 10.0.0 subnet). Named address which specify an entire network should begin with a trailing period (i.e. .proftpd.net for the entire proftpd.net domain).

## See also

Allow Order Limit

## Examples

```
<Limit LOGIN>
Order allow,deny
Allow from 128.44.26.,128.44.26.,myhost.mydomain.edu,.trusted-domain.org
```

```
Deny from all
</Limit>
```

# AllowAll

## Name

AllowAll –– Allow all clients

## Synopsis

**AllowAll** [ `AllowAll`]

`Default`
: Default is to implicitly AllowAll, but not explicitly

`Context`
: <Directory>, <Anonymous>, <Limit>, .ftpaccess

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

The AllowAll directive explicitly allows access to a <Directory>, <Anonymous> or <Limit> block. Although proftpd's default behavior is to allow access to a particular object, the default is an implicit allow. AllowAll creates an explicit allow, overriding any higher level denial directives.

## See also

DenyAll

## Examples

# AllowChmod

## Name

AllowChmod –– Enable the CHMOD command (deprecated)

## Synopsis

**AllowChmod** [ `on` | `off`]

`Default`
true
`Context`
server config, <Directory>, <Global>, <VirtualHost>, <Anonymous>, .ftpaccess
`Module`
mod_site
`Compatibility`
1.2.0rc1 and later –– Deprecated

## Description

This directive is deprecated, please use >Limit SITE_CHMOD< instead.

AllowChmod allows control over whether the "SITE CHMOD" command is allowed to clients.

## See also

## Examples

AllowChmod false

# AllowFilter

## Name

AllowFilter –– Regular expression of command arguments to be accepted

## Synopsis

**AllowFilter** [ `regular-expression`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0pre7 and later

## Description

AllowFilter allows the configuration of a regular expression that must be matched for all command arguments sent to ProFTPD. It is extremely useful in controlling what characters may be sent in a command to ProFTPD, preventing some possible types of attacks against ProFTPD. The regular expression is applied against the arguments to the command sent by the client, so care must be taken when creating a proper regex. Commands that fail the regex match result in a "Forbidden command" error being returned to the client. If the regular–expression argument contains whitespace, it must be enclosed in quotes.

## See also

DenyFilter

## Examples

```
# Only allow commands containing alphanumeric characters and whitespace
AllowFilter "^[a-zA-Z0-9 ,]*$"
```

# AllowForeignAddress

## Name

AllowForeignAddress –– Control the use of the PORT command

## Synopsis

**AllowForeignAddress** [ `on`|`off`]

`Default`
AllowForeignAddress off

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_core

`Compatibility`
1.1.7 and later

## Description

Normally, proftpd disallows clients from using the ftp PORT command with anything other than their own address (the source address of the ftp control connection), as well as preventing the use of PORT to specify a low–numbered (< 1024) port. In either case, the client is sent an "Invalid port" error and a message is syslog'd indicating either "address mismatch" or "bounce attack". By enabling this directive, proftpd will allow clients to transmit foreign data connection addresses that do not match the client's address. This allows such tricks as permitting a client to transfer a file between two FTP servers without involving itself in the actual data connection. Generally it's considered a bad idea, security–wise, to permit this sort of thing. AllowForeignAddress only affects data connection addresses; not tcp ports. There is no way (and no valid reason) to allow a client to use a low–numbered port in its PORT command.

## See also

## Examples

# AllowGroup

## Name

AllowGroup –– Group based allow rules

## Synopsis

**AllowGroup** [ `group–expression`]

`Default`
None
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
1.1.1 and later

## Description

AllowGroup specifies a group–expression that is specifically permitted within the context of the <Limit> block it is applied to. group–expression has the same format as that used in DefaultRoot, in that it should contain a comma separated list of groups or "not" groups (by prefixing a group name with the `!' character) that are to be allowed access to the block. The expression is parsed as a boolean "and" list, meaning that ALL elements of the expression must evaluate to logically true in order for the explicit allow to apply.

## See also

DenyGroup, DenyUser, AllowUser

## Examples

# Allow

## Name

AllowLogSymlinks –– Permit logging to symlinked files

## Synopsis

**AllowLogSymlinks** [ `"on"` | `"off"`]

`Default`
AllowLogSymlinks off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_log
`Compatibility`
1.2.2rc2 and later

## Description

By default, the server will the path of any configured SystemLog, any configured TransferLogs, and any configured ExtendedLogs to see if they are symbolic links. If the paths are symbolic links, the server will refuse to log to that link unless explicitly configured to do so via this directive.

### Security note:

Security note: this behaviour should not be allowed unless for a very good reason. By allowing the server to open symbolic links with its root privileges, you are allowing a potential symlink attack where the server could be tricked into overwriting arbitrary system files. You have been warned.

## See also

## Examples

```
AllowLogSymlinks on
```

# AllowOverwrite

## Name

AllowOverwrite –– Enable files to be overwritten

## Synopsis

**AllowOverwrite** [ `on` | `off`]

`Default`
AllowOverwrite off

`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>, .ftpaccess

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The AllowOverwrite directive permits newly transfered files to overwrite existing files. By default, ftp clients cannot overwrite existing files.

## See also

## Examples

# AllowRetrieveRestart

## Name

AllowRetrieveRestart –– Allow clients to resume downloads

## Synopsis

**AllowRetrieveRestart** [ `on` | `off`]

`Default`
AllowRetrieveRestart on
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>, .ftpaccess
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The AllowRetrieveRestart directive permits or denies clients from performing "restart" retrieve file transfers via the FTP REST command. By default this is enabled, so that clients may resume interrupted file transfers at a later time without losing previously collected data.

## See also

AllowStoreRestart

## Examples

# AllowStoreRestart

## Name

AllowStoreRestart –– Allow clients to resume uploads

## Synopsis

**AllowStoreRestart** [ `on`|`off`]

`Default`
AllowStoreRestart off

`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>, .ftpaccess

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The AllowStoreRestart directive permits or denies clients from "restarting" interrupted store file transfers (those sent from client to server). By default restarting (via the REST command) is not permitted when sending files to the server. Care should be taken to disallow anonymous ftp "incoming" transfers to be restarted, as this will allow clients to corrupt or increase the size of previously stored files (even if not their own).

The REST (Restart STOR) command is automatically blocked when HiddenStor is enabled, with the server returning a 501 error code to the client.

## See also

AllowRetrieveRestart DeleteAbortedStores HiddenStor

## Examples

# AllowUser

## Name

AllowUser –– User based allow rules

## Synopsis

**AllowUser** [ `user–expression`]

`Default`
None
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
1.1.7 and later

## Description

AllowUser specifies a user–expression that is specifically permitted access within the context of the <Limit> block it is applied to. user–expression has a similar syntax as that used in AllowGroup, in that it should contain a comma delimited list of users or "not" users (by prefixing a user name with the `!' character) that are to be allowed access to the block. The expression is parsed as a boolean "and" list, meaning that ALL elements of the expression must evaluate to logically true in order to the explicit allow to apply.

## See also

DenyUser DenyGroup AllowGroup

## Examples

# AnonRatio

## Name

AnonRatio –– Ratio directive

## Synopsis

**AnonRatio** [ `foo1 foo2 foo3`]

`Default`
None known
`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess
`Module`
mod_ratio
`Compatibility`
at least 1.2.0 and later

## Description

The AnonRatio directive ....

## See also

AnonRatio

## Examples

# AnonRequirePassword

## Name

AnonRequirePassword –– Make anonymous users supply a valid password

## Synopsis

**AnonRequirePassword** [ `on`|`off`]

`Default`
AnonRequirePassword off

`Context`
<Anonymous>

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

Normally, anonymous FTP logins do not require the client to authenticate themselves via the normal method of a transmitted cleartext password which is hashed and matched against an existing system user's password. Instead, anonymous logins are expected to enter their e–mail address when prompted for a password. Enabling the AnonRequirePassword directive requires anonymous logins to enter a valid password which must match the password of the user that the anonymous daemon runs as. However using AuthUsingAlias authentication can be matched against the password of the login username. This can be used to create "guest" accounts, which function exactly as normal anonymous logins do (and thus present a "chrooted" protected file system to the client), but require a valid password on the server's host system.

## See also

AnonymousGroup AuthAliasOnly AuthUsingAlias

## Examples

```
Example of a "guest" account configuration:
<Anonymous ~roger>
User roger
Group other
UserAlias proftpd roger
AnonRequirePassword on
# Deny write operations to all directories, underneath root-dir
# Default is to allow, so we don't need a <Limit> for read operations.
<Directory *>
<Limit WRITE>
```

```
DenyAll
</Limit>
</Directory>
# Deny all read/write operations in incoming. Because these are command-group
# limits, we can explicitly permit certain operations which will take precedence
# over our group limit.
<Directory incoming>
<Limit READ WRITE>
DenyAll
</Limit>
# The only command allowed in incoming is STOR (transfer file from client
to server)
<Limit STOR>
AllowAll
</Limit>
</Directory>
</Anonymous>
```

# Anonymous

## Name

Anonymous −− Define an anonymous server

## Synopsis

**Anonymous** [ `root−directory`]

`Default`
None
`Context`
server config,<VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The Anonymous configuration block is used to create an anonymous FTP login, and is terminated by a matching </Anonymous> directive. The root−directory parameters specifies which directory the daemon will first chdir to, and then chroot, immediately after login. Once the chroot operation successfully completes, higher level directories are no longer accessible to the running child daemon (and thus the logged in user). By default, proftpd assumes an anonymous login if the remote client attempts to login as the currently running user; unless the current user is root, in which case anonymous logins are not allowed regardless of the presence of an <Anonymous> block. To force anonymous logins to be bound to a user other than the current user, see the User and Group directives. In addition, if a User or Group directive is present in an <Anonymous> block, the daemon permanently switches to the specified uid/gid before chroot()ing. Normally, anonymous logins are not required to authenticate with a password, but are expected to enter a valid e−mail address in place of a normal password (which is logged). If this behavior is undesirable for a given <Anonymous> configuration block, it can be overridden via the AnonRequirePassword directive.

Note: Chroot()ed anonymous directories do not need to have supplemental system files in them, nor do they need to have any sort of specific directory structure. This is because proftpd is designed to acquire as much system information as possible before the chroot, and to leave open those files which are needed for normal operation and reside outside the new root directory.

## See also

## Examples

```
Example of a typical anonymous FTP configuration:
<Anonymous /home/ftp>
```

```
User ftp # After anonymous login, daemon runs as user ftp.
Group ftp # After anonymous login, daemon runs as group ftp.
UserAlias anonymous ftp # Client login as 'anonymous' is aliased to 'ftp'.
# Deny write operations to all directories, underneath root-dir
# Default is to allow, so we don't need a <Limit> for read operations.
<Directory *>
<Limit WRITE>
DenyAll
</Limit>
</Directory>
<Directory incoming>
<Limit READ WRITE>
DenyAll
</Limit>
<Limit STOR>
AllowAll
</Limit>
</Directory>
</Anonymous>
```

# AnonymousGroup

## Name

AnonymousGroup –– Treat group members as anonymous users

## Synopsis

**AnonymousGroup** [ `group–expression`]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
1.1.3 and later

## Description

The AnonymousGroup directive specifies a group–expression to which all matching users will be considered anonymous logins. The group–expression argument is a boolean logically ANDed list of groups to which the user must be a member of (or non–member if the group name is prefixed with a `!' character). For more information on group–expressions see the DefaultRoot directive. If the authenticating user is matched by an AnonymousGroup directive, no valid password is required, and a special dynamic anonymous configuration is created, with the user's home directory as the default root directory. If a DefaultRoot directive also applies to the user, this directory is used instead of the user's home dir. Great care should be taken when using AnonymousGroup, as improper configuration can open up user home directories to full read/write access to the entire world.

## See also

AuthAliasOnly AuthUsingAlias AnonRequirePassword DefaultRoot

## Examples

# AuthAliasOnly

## Name

AuthAliasOnly −− Allow only aliased login names

## Synopsis

**AuthAliasOnly** [ `on`|`off`]

`Default`
AuthAliasOnly off
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.1.3 and later

## Description

AuthAliasOnly restricts authentication to "aliased" logins only; i.e. those usernames provided by clients which are "mapped" to a real userid by the UserAlias directive. Turning AuthAliasOnly `on' in a particular context will cause proftpd to completely ignore all non−aliased logins for the entire context. If no contexts are available without AuthAliasOnly set to `on', proftpd rejects the client login and sends an appropriate message to syslog.

## See also

AnonymousGroup AuthUsingAlias AnonRequirePassword UserAlias

## Examples

# AuthGroupFile

## Name

AuthGroupFile −− Specify alternate group file

## Synopsis

**AuthGroupFile** [ `path`]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_unixpw
`Compatibility`
1.0.3/1.1.1 and later

## Description

AuthGroupFile specifies an alternate groups file, having the same format as the system /etc/group file, and if specified is used during authentication and group lookups for directory/access control operations. The path argument should be the full path to the specified file. AuthGroupFile can be configured on a per−VirtualHost basis, so that virtual FTP servers can each have their own authentication database (most often used in conjunction with AuthUserFile).

Note that this file need not reside inside a chroot()ed directory structure for Anonymous or DefaultRoot logins, as it is held open for the duration of client connections.

## See also

AuthUserFile

## Examples

# AuthPAM

## Name

AuthPAM –– Enable/Disable PAM authentication

## Synopsis

**AuthPAM** [ on|off]

Default
: on

Context
: server config,<VirtualHost>, <Global>

Module
: mod_pam

Compatibility
: 1.2.0rc1 and later

## Description

This directive determines whether PAM is used as an authentication method by ProFTPD. Enabled by default to fit in with the design policy of using PAM as the primary authentication mechanism.

## See also

## Examples

# AuthPAMAuthoritative

## Name

AuthPAMAuthoritative –– Set whether PAM is the authoritive authentication scheme

## Synopsis

**AuthPAMAuthoritative** [ `on` | `off`]

`Default`
off
`Context`
server config,<VirtualHost>, <Global>
`Module`
mod_pam
`Compatibility`
1.2.0pre3 and later

## Description

This directive allows you to control whether or not PAM is the ultimate authority on authentication. Setting this directive to on will cause authentication to fail if PAM authentication fails. The default setting, off, allows other modules and directives such as AuthUserFile and friends to authenticate users, should PAM authentication fail. If you are having problems with PAM and using other directives like AuthUserFile, set this directive to off.

## See also

## Examples

# AuthPAMConfig

## Name

AuthPAMConfig –– Select PAM service name

## Synopsis

**AuthPAMConfig** [ `service`]

Default
: ftp

Context
: server config,<VirtualHost>, <Global>

Module
: mod_pam

Compatibility
: 1.2.0rc1 and later

## Description

This directive allows you to specify the PAM service name used in authentication. PAM allows you to specify a service name to use when authenticating. This allows you to configure different PAM service names to be used for different virtual hosts. The directive was renamed from PAMConfig post 1.2.0 pre10.

## See also

## Examples

```
# Virtual host foobar authenticates differently than the rest

AuthPAMConfig foobar

# This assumes, that you have a PAM service named foobar
# configured in your /etc/pam.conf file or /etc/pam.d directory.
```

# AuthUserFile

## Name

AuthUserFile –– Specify alternate passwd file

## Synopsis

**AuthUserFile** [ `path`]

`Default`
None
`Context`
server config,<VirtualHost>, <Global>
`Module`
mod_unixpw
`Compatibility`
1.0.3/1.1.1 and later

## Description

AuthUserFile specifies an alternate passwd file, having the same format as the system /etc/passwd file, and if specified is used during authentication and user lookups for directory/access control operations. The path argument should be the full path to the specified file. AuthUserFile can be configured on a per−VirtualHost basis, so that virtual FTP servers can each have their own authentication database (most often used in conjunction with AuthGroupFile).

Note that this file need not reside inside a chroot()ed directory structure for Anonymous or DefaultRoot logins, as it is held open for the duration of client connections.

## See also

AuthGroupFile

## Examples

# AuthUsingAlias

## Name

AuthUsingAlias –– Authenticate via Alias–name instead of mapped username

## Synopsis

**AuthUsingAlias** [ `on`|`off`]

`Default`
AuthUsingAlias off

`Context`
<Anonymous>

`Module`
mod_core

`Compatibility`
1.2.0pre9 and later

## Description

AuthUsingAlias disables the resolving of mapped usernames for authentication purposes. For example, if you have mapped the username anonymous to the "real" user ftp, the password gets checked against the user "anonymous". When AuthUsingAlias is disabled, the checked username would be "ftp".

## See also

AnonymousGroup AuthAliasOnly AnonRequirePassword

## Examples

```
An example of an Anonymous configuration using
AuthUsingAlias
# Basic Read-Only Anonymous Configuration.
<Anonymous /home/ftp>
UserAlias             anonymous  nobody
UserAlias             ftp        nobody
AuthAliasOnly         on
<Limit WRITE>
DenyAll
</Limit>
</Anonymous>
# Give Full Read-Write Anonymous Access to certain users
<Anonymous /home/ftp>
AnonRequirePassword   on
AuthAliasOnly         on
AuthUsingAlias        on
# The list of authorized users.
```

```
# user/pass lookup is for each user, not password entry
# of server uid ('nobody' in this example).
UserAlias             fred       nobody
UserAlias             joe        nobody
<Limit ALL>
AllowAll
</Limit>
</Anonymous>
```

# Bind

## Name

Bind -- Bind the server or Virtualhost to a specific IP address

## Synopsis

**Bind** [ `IP address`]

`Default`
None
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
1.1.6 and later

## Description

The Bind directive allows additional IP addresses to be bound to a main or VirtualHost configuration. Multiple Bind directives can be used to bind multiple addresses. The address argument should be either a fully qualified domain name or a numeric dotted-quad IP address. Incoming connections destined to an additional address added by Bind are serviced by the context containing the directive. Additionally, if SocketBindTight is set to on, a specific listen connection is created for each additional address.

## See also

## Examples

# ByteRatioErrMsg

## Name

ByteRatioErrMsg –– Ratio directive

## Synopsis

**ByteRatioErrMsg** [ `foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The ByteRatioErrMsg directive .... Example: ByteRatioErrMsg

## See also

## Examples

# CDPath

## Name

CDPath –– Sets "search paths" for the cd command

## Synopsis

**CDPath** [ `directory`]

`Default`
: None

`Context`
: server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
: mod_core

`Compatibility`
: 1.2.0pre2 and later

## Description

Adds an entry to a search path that is used when changing directories. For example: CDPath /home/public CDPath /var/devel This allows a user to cd into any directory directly under /home/public or /var/devel, provided they have the appropriate rights. So, if /home/public/proftpd exists, cd proftpd will bring the user to that directory, regardless of where they currently are in the directory tree.

## See also

## Examples

# Class

## Name

Class –– Definition statements for class based tracking

## Synopsis

**Class** [ `"name" limit|regex|ip value`]

`Default`
None
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
1.2.0pre9 and later

## Description

Controls class based access. Class base access allows each connecting IP to be classified into a separate class. Each class has its own maximum number of connections. limit sets the maximum number of connections (default is 100) for that class name, regex sets a hostname regex (POSIX) for inclusion in the class and ip sets an IP/netmask based inclusion.

## See also

## Examples

Classes on
Class local limit 100
Class default limit 10
Class local regex .*foo.com
Class local ip 172.16.1.0/24

This creates two classes, local and default, with local being everything in *.foo.com and 172.16.1.* combined.

# Classes

## Name

Classes –– Enable Class based connection tracking

## Synopsis

**Classes** [ `on`|`off`]

`Default`
Off
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
1.2.0pre9 and later

## Description

Controls class based access. Enables class based access control. see: Class

## See also

## Examples

For examples, see Class

# CommandBufferSize

## Name

CommandBufferSize –– Limit the maximum command length

## Synopsis

**CommandBufferSize** [ `size`]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0pre7 and later

## Description

The CommandBufferSize directive controls the maximum command length permitted to be sent to the server. This allows you to effectively control what the longest command the server may accept it, and can help protect the server from various Denial of Service or resource–consumption attacks.

## See also

## Examples

# CwdRatioMsg

## Name

CwdRatioMsg –– Ratio directive

## Synopsis

**CwdRatioMsg** [ `foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The CwdRatioMsg directive .... Example: CwdRatioMsg

## See also

## Examples

# DefaultChdir

## Name

DefaultChdir –– Set starting directory for FTP sessions

## Synopsis

**DefaultChdir** [ `directory [group-expression]]`

`Default`
~
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_auth
`Compatibility`
1.2.0pre2 and later

## Description

Determines the directory a user is placed in after logging in. By default, the user is put in their home directory. The specified directory can be relative to the user's home directory. NOTE: if the specified directory is not available the user will not be able to log in.

## See also

DefaultRoot

## Examples

# DefaultQuota

## Name

DefaultQuota –– Sets the default quota

## Synopsis

**DefaultQuota** [ `value in bytes`]

`Default`
0
`Context`
server, <VirtualHost>, <Anonymous>
`Module`
mod_quota
`Compatibility`
at least 1.2.0 and later

## Description

The DefaultQuota directive sets the default quota in bytes, this value is used if the .quota file does not exist.

## See also

Quotas

## Examples

```
#
# Set default to 1kb
#
DefaultQuota 1024
```

# DefaultRoot

## Name

DefaultRoot –– Sets default chroot directory

## Synopsis

**DefaultRoot** [ `directory [group-expression]]`

Default
: DefaultRoot /

Context
: server config, <VirtualHost>, <Global>

Module
: mod_auth

Compatibility
: 0.99.0pl7 and later

## Description

The DefaultRoot directive controls the default root directory assigned to a user upon login. If DefaultRoot is set to a directory other than "/", a chroot operation is performed immediately after a client authenticates. This can be used to effectively isolate the client from a portion of the host system filespace. The specified root directory must begin with a / or can be the magic character '~'; meaning that the client is chroot jailed into their home directory.

If the DefaultRoot directive specifies a directory which disallows access to the logged–in user's home directory, the user's current working directory after login is set to the DefaultRoot instead of their normal home directory. DefaultRoot cannot be used in <Anonymous> configuration blocks, as the <Anonymous> directive explicitly contains a root directory used for Anonymous logins. The special character '~' is replaced with the authenticating user's home directory immediately after login. Note that the default root may be a subdirectory of the home directory, such as "~/anon–ftp".

The optional group–expression argument can be used to restrict the DefaultRoot directive to a unix group, groups or subset of groups. The expression takes the format: [!]group–name1[,[!]group–name2[,...]]. The expression is parsed in a logical boolean AND fashion, such that each member of the expression must evaluate to logically TRUE in order for the DefaultRoot directive to apply. The special character '!' is used to negate group membership.

Care should be taken when using DefaultRoot. Chroot "jails" should not be used as methods for implementing general system security as there are potentially ways that a user can "escape" the jail.

## See also

# Examples

```
Example of a DefaultRoot configuration:
ServerName "A test ProFTPD Server"
ServerType inetd
User ftp
Group ftp
#
# This causes proftpd to perform a chroot into the authenticating user's directory
immediately after login.
# Once this happens, the user is unable to "see" higher level directories.
# Because a group-expression is included, only users who are a member of
# the group 'users' and NOT a member of 'staff' will have their default
# root directory set to '~'.
DefaultRoot ~ users,!staff
...
```

# DefaultServer

## Name

DefaultServer –– Set the default server

## Synopsis

**DefaultServer** [ `on`|`off`]

`Default`
DefaultServer off

`Context`
server config,<VirtualHost>

`Module`
mod_core

`Compatibility`
0.99.0pl6 and later

## Description

The DefaultServer directive controls which server configuration is used as the default when an incoming connection is destined for an IP address which is neither the host's primary IP address or one of the addresses specified in a <VirtualHost> configuration block. Normally such "unknown" connections are issued a "no server available to service your request" message and disconnected. When DefaultServer is turned on for either the primary server configuration or a virtual server, all unknown destination connections are serviced by the default server. Only a single server configuration can be set to default.

## See also

## Examples

# DefaultTransferMode

## Name

DefaultTransferMode –– Set the default method of data transfer

## Synopsis

**DefaultTransferMode** [ `ascii` | `binary`]

`Default`
DefaultTransferMode ascii

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_core

`Compatibility`
1.2.0pre9 and later

## Description

DefaultTransferMode sets the default transfer mode of the server. By default, carriage–return/linefeed translation will be performed (ASCII mode).

## See also

## Examples

# DeferWelcome

## Name

DeferWelcome –– Don't show welcome message until user has authenticated

## Synopsis

**DeferWelcome** [ `DeferWelcome on|off`]

`Default`
DeferWelcome off

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The DeferWelcome directive configures a master or virtual server to delay transmitting the ServerName and address to new connections, until a client has successfully authenticated. If enabled, the initial welcome message will be exceedingly generic and will not give away any type of information about the host that the daemon is actively running on. This can be used by security–conscious administrators to limit the amount of "probing" possible from non–trusted networks/hosts.

## See also

ServerIdent ServerName

## Examples

# DeleteAbortedStores

## Name

DeleteAbortedStores –– Enable automatic deletion of partially uploaded files

## Synopsis

**DeleteAbortedStores** [ `DeleteAbortedStores on|off`]

`Default`
off
`Context`
server, <VirtualHost>, <Directory>, <Anonymous>, <Global>, .ftpaccess
`Module`
mod_core
`Compatibility`
1.2.0rc2 and later

## Description

The DeleteAbortedStores directive controls whether ProFTPD deletes partially uploaded files if the transfer is stopped via the ABOR command rather than a connection failure.

## See also

HiddenStor

## Examples

# Deny

## Name

Deny –– Access control directive

## Synopsis

**Deny** [ `Deny ["from"] "all"|"none"|host|network[,host|network[,...]]]`

Default
: None

Context
: <Limit>

Module
: mod_core

Compatibility
: 0.99.0pl6 and later

## Description

The Deny directive is used to create a list of hosts and/or networks which will explicitly be denied access to a given <Limit> context block. The magic keywords all and none can be used to indicate that all hosts are denied access, or that no hosts are explicitly denied (respectively). For more information on the syntax and usage of Deny see: Allow and Order.

## See also

Allow Order Limit

## Examples

# DenyAll

## Name

DenyAll -- Deny all clients

## Synopsis

**DenyAll** [ `DenyAll`]

`Default`
None
`Context`
<Directory>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The DenyAll directive is analogous to a combination of "order deny,allow <cr> deny from all", with the exception that it has a higher precendance when parsed. It is provided as a convenient method of completely denying access to a directory, anonymous ftp or limit block. Because of its precedence, it should not be intermixed with normal Order/Deny directives. The DenyAll directive can be overridden at a lower level directory by using AllowAll. DenyAll and AllowAll are mutually exclusive.

## See also

AllowAll

## Examples

# DenyFilter

## Name

DenyFilter –– Regular expression of command arguments to be blocked

## Synopsis

**DenyFilter** [ `DenyFilter regular-expression`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0pre7 and later

## Description

Similar to AllowFilter, DenyFilter specifies a regular expression which must not match any of the command arguments. If the regex does match, a "Forbidden command" error is returned to the client. This can be especially useful for forbidding certain command argument combinations from ever reaching ProFTPD.

*Notes:* The 'PASV' command cannot be blocked using this directive.

## See also

AllowFilter

## Examples

```
# We don't want to allow any commands with % being sent to the server
DenyFilter "%"
```

# DenyGroup

## Name

DenyGroup –– Group based deny rules

## Synopsis

**DenyGroup** [ `DenyGroup group–expression`]

`Default`
None
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
1.1.1 and later

## Description

DenyGroup specifies a group–expression that is specifically denied within the context of the <Limit> block it is applied to. group–expression has the same format as that used in DefaultRoot, in that it should contain a comma separated list of groups or "not" groups (by prefixing a group name with the `!' character) that are to be denied access to the block. The expression is parsed as a boolean "and" list, meaning that ALL elements of the expression must evaluate to logically true in order for the explicit deny to apply.

## See also

DenyUser, AllowUser AllowGroup

## Examples

# DenyUser

## Name

DenyUser –– User based deny rules

## Synopsis

**DenyUser** [ `DenyUser user–expression`]

`Default`
None
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
1.1.7 and later

## Description

DenyUser specifies a user–expression that is specifically denied within the context of the <Limit> block it is applied to. user–expression is a comma delimited list of users or "not" users (by prefixing a user name with the `!' character). The expression is parsed as a boolean "and" list, meaning that all elements of the expression must evaluate to logically true in order for the explicit deny to apply.

## See also

DenyGroup, AllowUser AllowGroup

## Examples

# Directory

## Name

Directory −− FIXME FIXME

## Synopsis

**Directory** [ `<Directory pathname>`]

`Default`
: None

`Context`
: server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

This directive creates a block of configuration directives which applies only to the specified directory and its sub−directories. The block is ended with </Directory>. Per−directory configuration is enabled during run−time with a "closest" match algorithm, meaning that the <Directory> directive with the closest matching path to the actual pathname of the file or directory in question is used. Per−directory configuration is inherited by all sub−directories until a closer matching <Directory> is encountered, at which time the original per−directory configuration is replaced with the closer match. Note that this does not apply to <Limit> </Limit> blocks, which are inherited by all sub−directories until a <Limit> block is reached in a closer match.

A trailing slash and wildcard ("/*") can be appended to the directory, specifying that the configuration block applies only to the contents (and sub−contents), not to the actual directory itself. Such wildcard matches always take precedence over non−wildcard <Directory> configuration blocks. <Directory> blocks cannot be nested (they are automatically nested at run−time based on their pathnames). Pathnames must always be absolute (except inside <Anonymous>), and should not reference symbolic links. Pathnames inside an <Anonymous> block can be relative, indicating that they are based on the anonymous root directory.

[Notes for ProFTPD 1.1.3 and later only] Pathnames that begin with the special character '~' and do not specify a username immediately after ~ are put into a special deferred mode. When in deferred mode, the directory context is not hashed and sorted into the configuration tree at boot time, but rather this hashing is deferred until a user authenticates, at which time the '~' character is replaced with the user's home directory. This allows a global <Directory> block which applies to all user's home directories, or sub−directories thereof.

## See also

Limit

# Examples

```
#Default usage of the directory directive
<Directory /users/robroy/private>
  HideNoAccess
</Directory>

#Example with username-expanding
<Directory ~/anon-ftp>
  <Limit WRITE>
    DenyAll
  </Limit>
</Directory>
```

# DirFakeGroup

## Name

DirFakeGroup –– Hide real file/directory group

## Synopsis

**DirFakeGroup** [ `DirFakeGroup On|Off [groupname]`]

`Default`
DirFakeGroup Off
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_ls
`Compatibility`
1.1.5

## Description

DirFakeGroup can be used to hide the true group of files (including directories, fifos, etc.) in a directory listing. If simply turned On, DirFakeGroup will display all files as being owned by group 'ftp'. Optionally, the groupname argument can be used to specify a specific group other than 'ftp'. "~" can be used as the argument in order to display the primary group name of the current user.

Both DirFakeGroup and DirFakeUser are completely cosmetic; the groupname or username specified don't need to exist on the system, and neither directive affects permissions, real ownership or access control in any way.

## See also

DirFakeUser DirFakeMode

## Examples

# DirFakeMode

## Name

DirFakeMode –– Hide real file/directory permissions

## Synopsis

**DirFakeMode** [ `DirFakeMode octal-mode`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_ls
`Compatibility`
1.1.6

## Description

The DirFakeMode directive configures a mode (or permissions) which will be displayed for ALL files and directories in directory listings. For each subset of permissions (user, group, other), the "execute" permission for directories is added in listings if the "read" permission is specified by this directive. As with DirFakeUser, and DirFakeGroup, the "fake" permissions shown in directory listings are cosmetic only, they do not affect real permissions or access control in any way.

## See also

DirFakeUser DirFakeGroup

## Examples

```
  DirFakeMode 0640

Will result in:

  -rw-r----- ... arbitrary.file
  drwxr-x--- ... arbitrary.directory
```

# DirFakeUser

## Name

DirFakeUser –– Hide real file/directory owner

## Synopsis

**DirFakeUser** [ `DirFakeUser On|Off [username]`]

`Default`
DirFakeUser Off

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_ls

`Compatibility`
1.1.5

## Description

DirFakeUser can be used to hide the true user owners of files (including directories, fifos, etc.) in a directory listing. If simply turned On, DirFakeUser will display all files as being owned by user 'ftp'. Optionally, the username argument can be used to specify a specific user other than 'ftp'. "~" can be used as the argument in order to display the current user's username.

Both DirFakeGroup and DirFakeUser are completely cosmetic; the groupname or username specified don't need to exist on the system, and neither directive affects permissions, real ownership or access control in any way.

## See also

DirFakeGroup DirFakeMode

## Examples

# DisplayConnect

## Name

DisplayConnect –– Sets connect banner file

## Synopsis

**DisplayConnect** [ `DisplayConnect filename`]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_code
`Compatibility`
1.2.0pre2 and later

## Description

The DisplayConnect directive configures an ASCII text filename which will be displayed to the user when they initially connect but before they login. The filename can be either relative or absolute. In the case of a relative filename, the file is searched for starting in the home directory of the user the server is running as. As this can lead confusion, absolute pathnames are suggested. If the file cannot be found or accessed, no error occurs and nothing is logged or displayed to the client.

## See also

## Examples

# DisplayFirstChdir

## Name

DisplayFirstChdir –– FIXME FIXME

## Synopsis

**DisplayFirstChdir** [ `DisplayFirstChdir filename`]

`Default`
None

`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>

`Module`
mod_code

`Compatibility`
0.99.0 and later, magic cookies only in 0.99.0pl10 and later

## Description

The DisplayFirstChdir directive configures an ASCII text filename which will be displayed to the user the first time they change into a directory (via CWD) per a given session. The file will also be displayed if proftpd detects that its last modification time has changed since the previous CWD into a given directory. If the filename is relative, it is looked for in the new directory that the user has changed into. Note that for anonymous ftp logins (see <Anonymous>), the file must reside inside the chroot()ed file system space. If the file cannot be found or accessed, no error occurs and nothing is logged or displayed to the client.

DisplayFirstChdir, DisplayConnect, DisplayLogin and DisplayQuit support the following "magic cookies" (only in 0.99.0pl10 and later), which are replaced with their respective strings before being displayed to the user.

```
%T      Current Time

%F      Available space on file system

%C      Current working directory

%R      Remote host name

%L      Local host name

%u      Username reported by ident protocol

%U      Username originally used in login

%M      Max number of connections

%N      Current number of connections

%E      Server admin's e-mail address
```

```
%x      The name of the user's class

%y      Current number of connections from the user's class

%z      Max number of connections from the user's class
```

NOTE: not all of these may have a rational value, depending on the context in which they're used (e.g., %u if ident lookups are off).

## See also

DisplayConnect DisplayLogin DisplayQuit

## Examples

# DisplayGoAway

## Name

DisplayGoAway –– FIXME FIXME

## Synopsis

**DisplayGoAway** [ `DisplayGoAway filename`]

`Default`
None

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_core

`Compatibility`
1.2.0pre8 and later

## Description

The DisplayGoAway directive specifies an ASCII text filename which will be displayed to the user if the class they're a member of has too many users logged in and their login request has been denied. DisplayGoAway supports the same "magic cookies" as DisplayFirstChdir.

## See also

DisplayFirstChdir

## Examples

# DisplayLogin

## Name

DisplayLogin –– FIXME FIXME

## Synopsis

**DisplayLogin** [ `DisplayLogin filename`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The DisplayLogin directive configures an ASCII text filename which will be displayed to the user when they initially login. The filename can be either relative or absolute. In the case of a relative filename, the file is searched for in the initial directory a user is placed in immediately after login (home directory for unix user logins, anonymous–root directory for anonymous logins). Note: that for jailed logins, the file must reside inside the chroot()ed file system space. If the file cannot be found or accessed, no error occurs and nothing is logged or displayed to the client. DisplayLogin supports the same "magic cookies" as DisplayFirstChdir.

## See also

## Examples

# DisplayQuit

## Name

DisplayQuit –– FIXME FIXME

## Synopsis

**DisplayQuit** [ `DisplayQuit filename`]

`Default`
None

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_core

`Compatibility`
1.2.0pre8 and later

## Description

DisplayQuit configures an ASCII text filename which will be displayed to the user when they quit. The filename can be either relative or absolute. In the case of a relative filename, the file is searched for in current directory a user is in when they logout –– for this reason, a absolute filename is usually preferable. NOTE: for jailed logins, the file must reside inside the chroot()ed file system space. If the file cannot be found or accessed, no error occurs and nothing is logged or displayed to the client. DisplayQuit supports the "magic cookies" listed under DisplayFirstChdir.

## See also

## Examples

# DisplayReadme

## Name

DisplayReadme –– FIXME FIXME

## Synopsis

**DisplayReadme** [ `DisplayReadme filename or pattern`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_readme
`Compatibility`
1.2.0pre8 and later

## Description

Module: mod_readme The DisplayReadme directive notifies the user of the last change date of the specified file or pattern. Only a single DisplayReadme directive is allowed per configuration scope. DisplayReadme README Will result in: Please read the file README it was last modified on Sun Oct 17 10:36:14 1999 – 0 days ago Being displayed to the user on a cwd. DisplayReadmePattern README* Will result in: Please read the file README it was last modified on Tue Jan 25 04:47:48 2000 – 0 days ago Please read the file README.first it was last modified on Tue Jan 25 04:48:04 2000 – 0 days ago Being displayed to the user on a cwd.

## See also

## Examples

# ExtendedLog

## Name

ExtendedLog −− FIXME FIXME

## Synopsis

**ExtendedLog** [ `filename [[command−classes] format−nickname]`]

`Default`
: None

`Context`
: server config, <VirtualHost>, <Anonymous> <Global>

`Module`
: mod_log

`Compatibility`
: 1.1.6pl1 and later

## Description

The ExtendedLog directive allows customizable logfiles to be generated, either globally or per VirtualHost. The filename argument must contain an absolute pathname to a logfile which will be appended to when proftpd starts; the pathname should not be to a file in a nonexistent directory, to a world−writeable directory, or be a symbolic link (unless AllowLogSymlinks is set to on). Multiple logfiles (potentially with different command classes and formats) can be created. Optionally, the command−classes argument can be used to control which types of commands are logged. If not command classes are specified, proftpd logs all commands by default (passwords are hidden). command−classes is a comma delimited (no whitespace!) list of which commands to log.

The following are valid classes: NONE No commands AUTH Authentication commands (USER, PASS) INFO Informational commands (PWD, SYST, etc) DIRS Directory commands (LIST, CWD, MKD, etc) READ File reading (RETR) WRITE File/directory writing or creation MISC Miscellaneous commands (SITE, etc) ALL All commands (default)

If a format−nickname argument is supplied, ExtendedLog will use the predefined logformat (created by LogFormat). Otherwise, the default format of "%h %l %u %t \"%r\" %s %b" is used.

## See also

AllowLogSymlinks, LogFormat, TransferLog

# Examples

For example, to log all read and write operations to /var/log/ftp.log (using the default format), you could:

```
ExtendedLog /var/log/ftp.log read,write
```

# FileRatioErrMsg

## Name

FileRatioErrMsg –– FIXME FIXME

## Synopsis

**FileRatioErrMsg** [ `FileRatioErrMsg foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The FileRatioErrMsg directive .... Example: FileRatioErrMsg

## See also

## Examples

# FooBarDirective

## Name

FooBarDirective –– FIXME FIXME

## Synopsis

**FooBarDirective** [ `FooBarDirective thingy`]

`Default`
none
`Context`
server config, <Anonymous>, <Limit>
`Module`
mod_sample
`Compatibility`
at least 1.2.0 and later

## Description

FooBarDirective is a dummy directive to be used as a coding example only.

## See also

## Examples

# Global

## Name

Global –– FIXME FIXME

## Synopsis

**Global** [ `<Global>`]

`Default`
None
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
1.1.6 and later

## Description

The Global configuration block is used to create a set of configuration directives which is applied universally to both the main server configuration and all VirtualHost configurations. Most, but not all other directives can be used inside a Global block. In addition, multiple <Global> blocks can be created. At runtime, all Global blocks are merged together and finally into each server's configuration. Global blocks are terminated by a matching </Global> directive.

## See also

## Examples

# Group

## Name

Group –– FIXME FIXME

## Synopsis

**Group** [ `Group groupid`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The Group directive configures which group the server daemon will normally run at. See User for more details.

## See also

## Examples

# GroupOwner

## Name

GroupOwner –– FIXME FIXME

## Synopsis

**GroupOwner** [ `GroupOwner groupname`]

`Default`
None
`Context`
<Anonymous>, <Directory>, .ftpaccess
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The GroupOwner directive configures which group all newly created directories and files will be owned by, within the context that GroupOwner is applied to. The group ID of groupname cannot be 0. Note that GroupOwner cannot be used to override the host OS/file system user/group paradigm. If the current user is not a member of the specified group, new files and directories will not be able to be chown()ed to the GroupOwner group. If this happens, file STOR (send file from client to server) and MKD/XMKD (mkdir) operations will succeed normally, however the new directory entries will be owned by the current user's default group (a warning message is also logged) instead of by the desired group. If you also use UserOwner in the same context, this restriction is lifted.

## See also

## Examples

# GroupPassword

## Name

GroupPassword –– FIXME FIXME

## Synopsis

**GroupPassword** [ `GroupPassword groupid hashed–password`]

`Default`
: None

`Context`
: server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
: mod_core

`Compatibility`
: 0.99.0pl5 and later

## Description

The GroupPassword directive creates a special "group" password which allows all users in the specified group to authenticate using a single password. The group/password supplied is only effective inside the context to which GroupPassword is applied. The hashed–password argument is a standard cleartext password which has been passed through the standard unix crypt() library function. Extreme care should be taken when using GroupPassword, as serious security problems may arise if group membership is not carefully controlled.

## See also

UserPassword

## Examples

# GroupRatio

## Name

GroupRatio -- Ratio directive

## Synopsis

**GroupRatio** [ `GroupRatio foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The GroupRatio directive .... Example: GroupRatio

## See also

## Examples

# HiddenStor

## Name

HiddenStor −− Enables more safe file uploads

## Synopsis

**HiddenStor** [ `HiddenStor on|off`]

`Default`
HiddenStor off
`Context`
<Directory>, <Anonymous>, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0pre5 and later

## Description

The HiddenStor directive enables two−step file uploads: files are uploaded as ".in.filename." and once the upload is complete, renamed to just "filename". This provides a degree of atomicity and helps prevent 1) incomplete uploads and 2) files being used while they're still in the progress of being uploaded. Note: if the temporary file name is already in use (e.g., a server crash during upload), it will prevent the file from being uploaded.

The REST (Restart STOR) command is automatically blocked when HiddenStor is enabled, with the server returning a 501 error code to the client.

## See also

AllowStoreRestart DeleteAbortedStores

## Examples

# HideGroup

## Name

HideGroup –– FIXME FIXME

## Synopsis

**HideGroup** [ `HideGroup groupid`]

`Default`
None
`Context`
<Directory>, <Anonymous>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The HideGroup directive configures a <Directory> or < Anonymous> block to hide all directory entries owned by the specified group, unless the group is the primary group of the currently logged–in, authenticated user . Normally, hidden directories and files cannot be seen via LIST or NLST commands but can be operated on via other FTP commands (CWD, DELE, RETR, etc). This behavior can be modified via the IgnoreHidden directive.

## See also

See Also: HideUser, HideNoAccess, IgnoreHidden

## Examples

# HideNoAccess

## Name

HideNoAccess –– Block the listing of directory entries to which the user has no access permissions

## Synopsis

**HideNoAccess** [ `HideNoAccess on|off`]

Default
: None

Context
: <Directory>,<Anonymous>

Module
: mod_core

Compatibility
: 0.99.0 and later

## Description

The HideNoAccess directive configures a <Directory> or <Anonymous> block to hide all directory entries in a directory listing (via the LIST or NLST FTP commands) to which the current logged–in, authenticated user has no access to. Normal Unix–style permissions always apply, so that although a user may not be able to see a directory entry that has HideNoAccess applied, they will receive a normal "Permission denied" error message when attempting to blindly manipulate the file system object. The directory or file can be made completely invisible to all FTP commands by applying IgnoreHidden in conjunction with HideNoAccess.

## See also

See Also: HideUser, HideGroup, IgnoreHidden

## Examples

# HideUser

## Name

HideUser –– FIXME FIXME

## Synopsis

**HideUser** [ `HideUser userid`]

`Default`
None
`Context`
<Directory>, <Anonymous>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The HideUser directive configures a <Directory> or <Anonymous> block to hide all directory entries owned by the specified user, unless the owning user is the currently logged–in, authenticated user. Normally, hidden directories and files cannot be seen via LIST or NLST commands but can be operated on via other FTP commands (CWD, DELE, RETR, etc). This behavior can be modified via the IgnoreHidden directive.

## See also

HideGroup, HideNoAccess, IgnoreHidden

## Examples

# HostRatio

## Name

HostRatio –– Ratio directive

## Synopsis

**HostRatio** [ `HostRatio foo1 foo2 foo3`]

`Default`
None known
`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess
`Module`
mod_ratio
`Compatibility`
at least 1.2.0 and later

## Description

The HostRatio directive .... Example: HostRatio

## See also

## Examples

# IdentLookups

## Name

IdentLookups –– Toggle ident lookups

## Synopsis

**IdentLookups** [ `IdentLookups on|off`]

`Default`
: IdentLookups on

`Context`
: server config, <VirtualHost>, <Global>

`Module`
: mod_core

`Compatibility`
: 1.1.5 and later

## Description

Normally, when a client initially connects to proftpd, the ident protocol (RFC1413) is used to attempt to identify the remote username. This can be controlled via the IdentLookups directive.

## See also

## Examples

# IgnoreHidden

## Name

IgnoreHidden –– Treat 'hidden' files as if they don't exist

## Synopsis

**IgnoreHidden** [ `IgnoreHidden on|off`]

`Default`
IgnoreHidden off
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

Normally, files hidden via HideNoAccess, HideUser or HideGroup can be operated on by all FTP commands (assuming Unix file permissions allow access), even though they do not appear in directory listings. Additionally, even when normal file system permissions disallow access, proftpd returns a "Permission denied" error to the client, indicating that the requested object does exist, even if it cannot be acted upon. IgnoreHidden configures a <Limit> block to completely ignore any hidden directory entries for the set of limited FTP commands. This has the effect of returning an error similar to "No such file or directory" when the client attempts to use the limited command upon a hidden directory or file.

## See also

## Examples

# Include

## Name

Include –– Load additional configuration directives from a file

## Synopsis

**Include** [ `Include file`]

`Default`
None
`Context`
server config, <Directory>, <Anonymous>, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0 and later

## Description

This directive allows you to include another configuration file within your current configuration file. The given file argument must be the full path to the file to be included.

## See also

## Examples

# LDAPAuthBinds

## Name

LDAPAuthBinds –– FIXME FIXME

## Synopsis

**Syntax: LDAPAuthBinds** [ `on` `off` ]

**FIX FIX FIX**

`Default`
: LDAPAuthBinds off in mod_ldap <= 2.7.6, LDAPAuthBinds on in mod_ldap >= 2.8

`Context`
: server config, <VirtualHost>, <Global>

`Module`
: mod_ldap

`Compatibility`
: mod_ldap v2.5 and later

## Description

By default, the DN specified by LDAPDNInfo will be used to bind to the LDAP server to obtain user information, including the userPassword attribute. If LDAPAuthBinds is set to on, the DN specified by LDAPDNInfo will be used to fetch all user information except the userPassword attribute. Then, mod_ldap will bind to the LDAP server as the user who is logging in via FTP with the user–supplied password. If this bind succeeds, the user is considered authenticated and is allowed to log in. This method of LDAP authentication has the added benefit of supporting any password encryption scheme that your LDAP server supports.

## See also

## Examples

# LDAPDefaultAuthScheme

## Name

LDAPDefaultAuthScheme –– Set the authentication scheme/hash that is used when no leading {hashname} is present.

## Synopsis

**LDAPDefaultAuthScheme** [ `crypt` `clear` ]

`Default`
LDAPDefaultAuthScheme "crypt"
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.0 and later

## Description

Specifies the authentication scheme used for passwords with no {prefix} in the LDAP database. For example, if you are using something like userPassword: mypass in your LDAP database, you would want to set LDAPDefaultAuthScheme to clear.

## See also

## Examples

# LDAPDefaultGID

## Name

LDAPDefaultGID –– Set the default GID to be assigned to users when no uidNumber attribute is found.

## Synopsis

**LDAPDefaultGID** [ `default-gid` ]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.0 and later

## Description

This directive is useful primarily in virtual–user environments common in large–scale ISPs and hosting organizations. If a user does not have a LDAP gidNumber attribute, the LDAPDefaultGID is used. This allows one to have a large number of users in an LDAP database without gidNumber attributes; setting this configuration directive will automatically assign those users a single GID.

## See also

## Examples

# LDAPDefaultUID

## Name

LDAPDefaultUID −− Set the default GID to be assigned to users when no uidNumber attribute is found.

## Synopsis

**LDAPDefaultUID** [ `default-uid` ]

`Default`
: None

`Context`
: server config, <VirtualHost>, <Global>

`Module`
: mod_ldap

`Compatibility`
: mod_ldap v2.0 and later

## Description

This directive is useful primarily in virtual−user environments common in large−scale ISPs and hosting organizations. If a user does not have a LDAP uidNumber attribute, the LDAPDefaultUID is used. This allows one to have a large number of users in an LDAP database without uidNumber attributes; setting this configuration directive will automatically assign those users a single UID.

## See also

## Examples

# LDAPDNInfo

## Name

LDAPDNInfo –– Set DN information to be used for initial bind

## Synopsis

**LDAPDNInfo** [ `LDAPDNInfo "ldap-dn" "dn-password"` ]

`Default`
LDAPDNInfo "" "" (anonymous bind)

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_ldap

`Compatibility`
mod_ldap v2.0 and later

## Description

This directive specifies the LDAP DN and password to use when binding to the LDAP server. If this configuration directive is not specified, anonymous binds are used.

## See also

## Examples

# LDAPDoAuth

## Name

LDAPDoAuth -- Enable LDAP authentication

## Synopsis

**LDAPDoAuth** [ `on off` ] [ `"auth-base-prefix"` ] [ `"search-filter-template"` ]

`Default`
LDAPDoAuth off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.0 and later

## Description

This configuration directive activates LDAP authentication. The second argument to this directive is the LDAP prefix to use for authentication. The third argument is a template to be used for the search filter; %u will be replaced with the username that is being authenticated. By default, the search filter template "(&(uid=%u)(objectclass=posixAccount))" is used. Search filter templates are only supported in mod_ldap v2.7 and later.

## See also

## Examples

# LDAPDoGIDLookups

## Name

LDAPDoGIDLookups −− Enable LDAP lookups for user group membership and GIDs in directory listings

## Synopsis

**LDAPDoGIDLookups** [ `on` `off` ] [ `"uid-base-prefix"` ] [ `"search-filter-template"` ]

`Default`
LDAPDoGIDLookups off

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_ldap

`Compatibility`
mod_ldap v2.0 and later

## Description

This configuration directive activates LDAP GID−to−name lookups in directory listings. The second argument to this directive is the LDAP prefix to use for GID−to−name lookups. The third argument is a template to be used for the search filter; %u will be replaced with the GID that is being looked up. By default, the search filter template "(&(gidNumber=%u)(objectclass=posixGroup))" is used. Search filter templates are only supported in mod_ldap v2.7 and later.

## See also

## Examples

# LDAPDoUIDLookups

## Name

LDAPDoUIDLookups −− Enable LDAP lookups for UIDs in directory listings

## Synopsis

**LDAPDoUIDLookups** [ `on off` ] [ `"search−filter−template"` ] [ `"uid−base−prefix"` ]

`Default`
LDAPDoUIDLookups off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.0 and later

## Description

This configuration directive activates LDAP UID−to−name lookups in directory listings. The second argument to this directive is the LDAP prefix to use for UID−to−name lookups. The third argument is a template to be used for the search filter; %u will be replaced with the UID that is being looked up. By default, the search filter template "(&(uidNumber=%u)(objectclass=posixAccount))" is used. Search filter templates are only supported in mod_ldap v2.7 and later.

## See also

## Examples

# LDAPForceDefaultGID

## Name

LDAPForceDefaultGID –– Force all LDAP–authenticated users to use the same GID.

## Synopsis

**Syntax: LDAPForceDefaultGID** [ `on` `off` ]

`Default`
LDAPForceDefaultGID off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.8 and later

## Description

Even when a LDAPDefaultGID is configured, mod_ldap will allow individual users to have gidNumber attributes that will override this default GID. With LDAPForceDefaultGID enabled, all LDAP–authenticated users are given the default GID; GIDs may not be overridden by gidNumber attributes.

## See also

## Examples

# LDAPForceDefaultUID

## Name

LDAPForceDefaultUID −− Force all LDAP−authenticated users to use the same UID.

## Synopsis

**Syntax: LDAPForceDefaultUID** [ `on` `off` ]

Default
: LDAPForceDefaultUID off

Context
: server config, <VirtualHost>, <Global>

Module
: mod_ldap

Compatibility
: mod_ldap v2.8 and later

## Description

Even when a LDAPDefaultUID is configured, mod_ldap will allow individual users to have uidNumber attributes that will override this default UID. With LDAPForceDefaultUID enabled, all LDAP−authenticated users are given the default UID; UIDs may not be overridden by uidNumber attributes.

## See also

## Examples

# LDAPHomedirOnDemand

## Name

LDAPHomedirOnDemand −− Enable the creation of user home directories on demand

## Synopsis

**LDAPHomedirOnDemand** [ `on off` ] [ `directory−mode` ]

`Default`
LDAPHomedirOnDemand off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.0 and later

## Description

LDAPHomedirOnDemand activates on−demand home directory creation. If a user logs in and does not yet have a home directory, a home directory is created automatically.

In mod_ldap <= 2.7.6, the home directory will be owned by the same user and group that ProFTPD runs as (see the User and Group configuration directives). mod_ldap >= 2.8 can create home directories for users with any UID/GID, not just those with the same UID/GID as the main ProFTPD server.

The second argument allows you to specify the mode (default permissions) to use when creating home directories on demand, subject to ProFTPD's umask (see the Umask directive). If no directory mode is specified, the default of 0755 is used. Directory mode setting is only supported in mod_ldap v2.7 or later.

## See also

## Examples

# LDAPHomedirOnDemandPrefix

## Name

LDAPHomedirOnDemandPrefix −− Enable the creation of user home directories on demand

## Synopsis

**LDAPHomedirOnDemandPrefix** [ `leading-path` ]

`Default`
LDAPHomedirOnDemandPrefix off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.8 and later

## Description

LDAPHomedirOnDemandPrefix enables a prefix to be specified for on−demand home directory creation. This is most useful if mod_ldap is being used to authenticate against an LDAP directory that does not return a homeDirectory attribute, either because it cannot (Microsoft Active Directory, for example) or because you do not wish to extend your existing directory schema.

For example, setting this directive to "/home" and logging in as the user "joe" would result in his home directory being created as "/home/joe". The directory will be created with the mode specified in LDAPHomedirOnDemand. To use this directive, LDAPHomedirOnDemand must be enabled.

## See also

## Examples

# LDAPHomedirOnDemandPrefixNoUsername

## Name

LDAPHomedirOnDemandPrefixNoUsername –– FIXFIXFIX

## Synopsis

**LDAPHomedirOnDemandPrefixNoUsername** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_ldap
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# LDAPHomedirOnDemandSuffix

## Name

LDAPHomedirOnDemandSuffix — Specify an additional directory to be created inside a user's home directory on demand.

## Synopsis

**LDAPHomedirOnDemandSuffix** [ `additional-directory1` `additional-directory2` `additional-directory3` ]

Default
: LDAPHomedirOnDemandSuffix ""

Context
: server config, <VirtualHost>, <Global>

Module
: mod_ldap

Compatibility
: mod_ldap v2.6 and later.

## Description

to be created within a user's home directory when it is created on demand. For example, if a user's home directory is "/home/user", setting this configuration directive to "public_html" will also create "/home/user/public_html" on demand. In mod_ldap v2.7.6 and earlier, you must also activate LDAPHomedirOnDemand in your configuration.

mod_ldap >= 2.8 supports multiple suffix arguments and does not require LDAPHomedirOnDemand to be enabled.

## See also

## Examples

# LDAPNegativeCache

## Name

LDAPNegativeCache –– Enable negative caching for LDAP lookups

## Synopsis

**LDAPNegativeCache** [ `on` `off` ]

`Default`
LDAPNegativeCache off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v1.1 and later

## Description

LDAPNegativeCache specifies whether or not to cache negative responses from the LDAP server when using LDAP for UID/GID lookups. This option is useful if you also use/are in transition from another authentication system; if there are many users in your old authentication system that aren't in the LDAP database, there can be a significant delay when a directory listing is performed as the UIDs not in the LDAP database are repeatedly looked up in an attempt to present usernames instead of UIDs in directory listings. With LDAPNegativeCache set to on, negative ("not found") responses from the LDAP server will be cached and speed will improve on directory listings that contain many users not present in the LDAP database.

## See also

## Examples

# LDAPQueryTimeout

## Name

LDAPQueryTimeout –– Set a timeout for LDAP queries

## Synopsis

**LDAPQueryTimeout** [ `timeout-seconds` ]

`Default`
LDAPQueryTimeout default–api–timeout

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_ldap

`Compatibility`
mod_ldap v2.0 and later

## Description

Sets the timeout used for LDAP directory queries. The default is the default timeout used by your LDAP API.

## See also

## Examples

# LDAPSearchScope

## Name

LDAPSearchScope –– Specify the search scope used in LDAP queries

## Synopsis

**LDAPSearchScope** [ `onelevel` `subtree` ]

`Default`
LDAPSearchScope subtree

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_ldap

`Compatibility`
mod_ldap v2.6 and later

## Description

Set the scope used for LDAP searches. The default setting, subtree, searches for all entries in the tree from the current level down. Setting this directive to onelevel searches only one level deep in the LDAP tree.

## See also

## Examples

# LDAPServer

## Name

LDAPServer –– Specify the LDAP server to use for lookups

## Synopsis

**LDAPServer** [ `"hostname1:port1 hostname2:port2"` ]

`Default`
LDAPServer "localhost"

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_ldap

`Compatibility`
mod_ldap v1.0 and later

## Description

LDAPServer allows you to to specify the hostname(s) and port(s) of the LDAP server(s) to use for LDAP authentication. If no LDAPServer configuration directive is present, the default LDAP servers specified by your LDAP API will be used.

## See also

## Examples

# LDAPUseTLS

## Name

LDAPUseTLS –– Enable TLS/SSL connections to the LDAP server.

## Synopsis

**Syntax: LDAPUseTLS** [ `on off` ]

`Default`
LDAPUseTLS off
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_ldap
`Compatibility`
mod_ldap v2.8 and later

## Description

By default, mod_ldap connects to the LDAP server via a non–encrypted connection. Enabling this option causes mod_ldap to use an encrypted (TLS/SSL) connection to the LDAP server. If a secure connection to the LDAP server fails, mod_ldap will not authenticate users (mod_ldap will *not* fall back to an unsecure connection).

## See also

## Examples

# LeechRatioMsg

## Name

LeechRatioMsg –– Sets the 'over ratio' error message

## Synopsis

**LeechRatioMsg** [ `LeechRatioMsg foo1 foo2 foo3`]

`Default`
None known
`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess
`Module`
mod_ratio
`Compatibility`
at least 1.2.0 and later

## Description

The LeechRatioMsg directive defines the response message sent back to the client upon breaking their quota limits.

## See also

## Examples

```
LeechRatioMsg "please upload as well as download"
```

# Limit

## Name

Limit –– FIXME FIXME

## Synopsis

**Limit** [ `<Limit command|command-group [command2 ..]>`]

`Default`
None
`Context`
server config, <VirtualHost>, <Directory>, <Anonymous>, <Global>, .ftpaccess
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The Limit configuration block is used to place access restrictions on one or more FTP commands, within a given context. Limits flow downward, so that a Limit configuration in the server config context applies to all <Directory> and <Anonymous> blocks that also reside in the configuration; until it is overridden by a "lower" <Limit> block. Any number of command parameters can be specified, against which the contents of the <Limit> block will be applied. command can be any valid FTP command, but is generally one of the following: CWD (Change Working Directory) Sent by client when changing directories. MKD / XMKD (MaKe Directory) Sent by client to create a new directory. RNFR (ReName FRom), RNTO (ReName TO) Sent as a pair by client to rename a directory entry. DELE (DELEte) Sent by client to delete a file. RMD / XRMD (ReMove Directory) Sent by client to remove a directory. RETR (RETRieve) Transfer a file from the server to the client. STOR (STORe) Transfer a file from the client to the server. In addition, the following command–groups are accepted. They have a lower precedence than real commands, meaning that a real command limit will always be applied instead of the command–group. READ All FTP commands which deal with file reading (directory listing not included): RETR, SITE, SIZE, STAT WRITE All FTP commands which deal with file or directory write/creation/deletion: APPE, DELE, MKD, RMD, RNTO, STOR, XMKD, XRMD DIRS All FTP commands which deal with directory listing: CDUP, CWD, LIST, MDTM, NLST, PWD, RNFR, XCUP, XCWD, XPWD ALL ALL FTP commands (identical to READ WRITE DIRS). Note this group has the lowest precedence of all; it will not override a limit imposed by another command–group (e.g. DIRS). Finally, a special command is allowed which can be used to control login access: LOGIN Connection or login to the server. Applying a <Limit> to this pseudo–command can be used to allow or deny initial connection or login to the context. It has no effect, and is ignored, when used in a context other than server config, <VirtualHost> or <Anonymous> (i.e. using it in a <Directory> context is meaningless). <Limit> command restrictions should not be confused with file/directory access permission. While limits can be used to restrict a command on a certain directory, they cannot be used to override the file permissions inherent to the base operating/file system. The following FTP commands cannot be restricted via <Limit>: ABOR HELP MODE (not implemented, always S) NOOP PASS (use <Limit LOGIN>) PASV PORT QUIT REST (use AllowRetrieveRestart, AllowStoreRestart) STRU (not implemented, always F) SYST TYPE USER (use <Limit LOGIN>)

## See also

See Also: IgnoreHidden

## Examples

# LogFormat

## Name

LogFormat –– Specify a logging format

## Synopsis

**LogFormat** [ `LogFormat nickname "format-string"`]

`Default`
: LogFormat default "%h %l %u %t \"%r\" %s %b"

`Context`
: server config

`Module`
: mod_log

`Compatibility`
: 1.1.6pl1 and later

## Description

The LogFormat directive can be used to create a custom logging format for use with the ExtendedLog directive. Once created, the format can be referenced by the specified nickname. The format−string argument can consist of any combination of letters, numbers and symbols. The special character % is used to start a meta−sequence (see below). To insert a literal % character, use %%.

The following meta sequences are available and are replaced as indicated when logging. %A Anonymous username (password given), or UNKNOWN if non−anonymous %b Bytes sent for request %f Filename stored or retrieved, absolute path (not chrooted) %F Filename stored or retrieved, as the client sees it %{FOOBAR}e Contents of environment variable FOOBAR. Note that the server does not set any environment variables itself. %h Remote host name %a Remote IP address %l Remote username (from ident), or UNKNOWN if ident lookup failed %m Command (method) name received from client, e.g., RETR %p Local server port number %v Local server name %P Local server process id (pid) %r Full command line received from client %t Current local time %{format}t Current local time formatted (strftime(3) format) %T Time taken to transmit/receive file, in seconds %s Numeric FTP response code (status) %u Local authenticated userid

## See also

ExtendedLog, TransferLog

## Examples

# LoginPasswordPrompt

## Name

LoginPasswordPrompt –– FIXME FIXME

## Synopsis

**LoginPasswordPrompt** [ `LoginPasswordPrompt on|off`]

`Default`
LoginPasswordPrompt on

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_auth

`Compatibility`
1.2.0pre1 and later

## Description

If set to off, ProFTPd will skip the password request if the login will be denied regardless of password, e.g., if a <Limit LOGIN> directive forbids the connection.

## See also

## Examples

# LsDefaultOptions

## Name

LsDefaultOptions –– FIXME FIXME

## Synopsis

**LsDefaultOptions** [ `LsDefaultOptions "options string"`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_ls
`Compatibility`
1.1.6 and later

## Description

Normally, FTP commands involving directory listings (NLST, LIST and STAT) use the arguments (options) passed by the client to determine what files are displayed and the format they are displayed in. Using the LsDefaultOptions directive can alter the default behavior of such listings, but implying that a certain option (or options) is always present. For example, to force all directory listings to always display ".dotfiles", one might: LsDefaultOptions "−a"

## See also

## Examples

# MasqueradeAddress

## Name

MasqueradeAddress –– Configure the server address presented to clients

## Synopsis

**MasqueradeAddress** [ `MasqueradeAddress ip-address|dns-hostname`]

`Default`
none
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
1.2.2 and later

## Description

MasqueradeAddress causes the server to display the network information for the specified IP address or DNS hostname to the client, on the assumption that that IP address or DNS host is acting as a NAT gateway or port forwarder for the server.

## See also

## Examples

```
MasqueradeAddress nat-gw.mydomain.com
```

# MaxClients

## Name

MaxClients –– Limits the number of users that can connect

## Synopsis

**MaxClients** [ `MaxClients number|none [message]`]

`Default`
MaxClients none

`Context`
server config, <Anonymous>, <VirtualHost>, <Global>

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The MaxClients directive configures the maximum number of authenticated clients which may be logged into a server or anonymous account. Once this limit is reached, additional clients attempting to authenticate will be disconnected. The special value none may be supplied which removes all maximum connection limits from the applicable configuration context. Additionally, an optional message argument may be used which will be displayed to a client attempting to exceed the maximum value; immediately before disconnection. The message argument is parsed for the magic string "%m", which is replaced with the configured maximum value. If message is not supplied, a system–wide default message is used. Example: MaxClients 5 "Sorry, the maximum number of allowed users are already connected (%m)" Results in: 530 Sorry, the maximum number of allowed users are already connected (5)

## See also

## Examples

# MaxClientsPerHost

## Name

MaxClientsPerHost –– Limits the connections per client machine

## Synopsis

**MaxClientsPerHost** [ `MaxClientsPerHost number|none [message]`]

Default
: MaxClientsPerHost none

Context
: server config, <Anonymous>, <VirtualHost>, <Global>

Module
: mod_core

Compatibility
: 1.1.7 and later

## Description

The MaxClientsPerHost directive configures the maximum number of clients allowed to connect per host. The optional argument message may be used which will be displayed to a client attempting to exceed the maximum value. If message is not supplied, a default message of "Sorry, the maximum number clients (%m) from your host are already connected." is used.

## See also

MaxClients, MaxHostsPerUser

## Examples

```
MaxClientsPerHost 1 "Sorry, you may not connect more than one time."
Results in: 530 Sorry, you may not connect more than one time.
```

# MaxHostsPerUser

## Name

MaxHostsPerUser –– Limit the number of connections per userid

## Synopsis

**MaxHostsPerUser** [ `MaxHostsPerUser number|none [message]`]

Default
: MaxHostsPerUser none

Context
: server config, <Anonymous>, <VirtualHost>, <Global>

Module
: mod_core

Compatibility
: 1.2.4 and later

## Description

The MaxHostsPerUser directive configures the maximum number of times any given login can connection at any given time. The optional argument message may be used which will be displayed to a client attempting to exceed the maximum value. If message is not supplied, a default message of "Sorry, the maximum number of hosts (%m) for this user already connected."

## See also

MaxClients, MaxClientsPerHost

## Examples

```
MaxHostsPerUser 1 "Sorry, you may not connect more than one time."
Results in: 530 Sorry, you may not connect more than one time.
```

# MaxInstances

## Name

MaxInstances –– Sets the maximum number of child processes to be spawned

## Synopsis

**MaxInstances** [ `MaxInstances number`]

`Default`
MaxInstances none
`Context`
server config
`Module`
mod_core
`Compatibility`
1.1.6pl1

## Description

The MaxInstances directive configures the maximum number of child processes that may be spawned by a parent proftpd process in standalone mode. The directive has no effect when used on a server running in inetd mode. Because each child proftpd process represents a single client connection, this directive also controls the maximum number of simultaneous connections allowed. Additional connections beyond the configured limit are syslog'd and silently disconnected. The MaxInstances directive can be used to prevent undesirable denial–of–service attacks (repeatedly connecting to the ftp port, causing proftpd to fork–bomb). By default, no limit is placed on the number of child processes that may run at one time.

## See also

## Examples

# MaxLoginAttempts

## Name

MaxLoginAttempts –– Sets how many password attempts are allowed before disconnection

## Synopsis

**MaxLoginAttempts** [ `MaxLoginAttempts number`]

`Default`
MaxLoginAttempts 3
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The MaxLoginAttempts directive configures the maximum number of times a client may attempt to authenticate to the server during a given connection. After the number of attempts exceeds this value, the user is disconnected and an appropriate message is logged via the syslog mechanism.

## See also

## Examples

# MultilineRFC2228

## Name

MultilineRFC2228 −− FIXME FIXME

## Synopsis

**MultilineRFC2228** [ `MultilineRFC2228 on|off`]

`Default`
MultilineRFC2228 off

`Context`
server config

`Module`
mod_core

`Compatibility`
1.2.0pre3 and later

## Description

By default, proftpd sends multiline responses as per RFC 959, i.e.: 200−First line More lines... 200 Last line RFC 2228 specifies that "6xy" response codes will be sent as follows: 600−First line 600−More lines... 600 Last line Note that 2228 ONLY specifies this for response codes starting with '6'. Enabling this directive causes ALL responses to be sent in this format, which may be more compatible with certain web browsers and clients. Also note that this is NOT the same as wu−ftpd's multiline responses, which do not comply with any RFC. Using this method of multilines is more likely to be compatible with all clients, although it isn't strictly RFC, and is thus not enabled by default.

## See also

## Examples

# MySQLInfo

## Name

MySQLInfo –– Configures the MySQL driver

## Synopsis

**MySQLInfo** [ `hostname`] [ `sqluser`] [ `sqlpass`] [ `dbname`]

`Default`
none

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0rc2 and later

## Description

This directive is deprecated as of 1.2.0. Please use SQLConnectInfo instead.

Configures the MySQL database driver (the database may be remote). A connection isn't made until use of a SQL feature requires it, after which it may be held open for the lifetime of the FTP session depending on the directives in use. Use `"" to specify a null password.

## See also

## Examples

# Order

## Name

Order –– Configures the precedence of the Limit directives

## Synopsis

**Order** [ `Order allow,deny|deny,allow`]

`Default`
Order allow,deny
`Context`
<Limit>
`Module`
mod_core
`Compatibility`
0.99.0pl6 and later

## Description

The Order directive configures the order in which Allow and Deny directives are checked inside of a <Limit> block. Because Allow directives are permissive, and Deny directives restrictive, the order in which they are examined can significantly alter the way security functions. If the default setting of allow,deny is used, "allowed" access permissions are checked first. If an Allow directive explicitly allows access to the <Limit> context, access is granted and any Deny directives are never checked. If Allow did not explicitly permit access, Deny directives are checked. If any Deny directive applies, access is explicitly denied. Otherwise, access is granted. When deny,allow is used, "deny" access restrictions are checked first. If any restriction applies, access is denied immediately. If nothing is denied, Allow permissions are checked. If an Allow explicitly permits access, access to the entire context is permitted; otherwise access is implicitly denied. For clarification, the following illustrates the steps used when checking Allow/Deny access: Order allow,deny Check Allow directives. If one or more apply, exit with result: ALLOW Check Deny directives. If one or more apply, exit with result: DENY Exit with default implicit ALLOW Order deny,allow Check Deny directives. If one or more apply, exit with result: DENY Check Allow directives. If one or more apply, exit with result: ALLOW Exit with default implicit: DENY

## See also

## Examples

# PassivePorts

## Name

PassivePorts –– Specify the ftp–data port range to be used

## Synopsis

**PassivePorts** [ `PassivePorts min–pasv–port max–pasv–port`]

`Default`
None
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0rc2 and later

## Description

PassivePorts restricts the range of ports from which the server will select when sent the PASV command from a client. The server will randomly choose a number from within the specified range until an open port is found. Should no open ports be found within the given range, the server will default to a normal kernel–assigned port, and a message logged.

The port range selected must be in the non–privileged range (eg. greater than or equal to 1024); it is STRONGLY RECOMMENDED that the chosen range be large enough to handle many simultaneous passive connections (for example, 49152–65534, the IANA–registered ephemeral port range).

## See also

## Examples

```
# Use the IANA registered ephemeral port range
PassivePorts 49152 65534
```

# PathAllowFilter

## Name

PathAllowFilter –– FIXME FIXME

## Synopsis

**PathAllowFilter** [ `PathAllowFilter regular-expression`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.1.7 and later

## Description

PathAllowFilter allows the configuration of a regular expression that must be matched for all newly uploaded (stored) files. The regular expression is applied against the entire pathname specified by the client, so care must be taken when creating a proper regex. Paths that fail the regex match result in a "Forbidden filename" error being returned to the client. If the regular–expression argument contains whitespace, it must be enclosed in quotes.

## See also

## Examples

```
# Only allow a-z 0-9 . - _ in file names,
PathAllowFilter ^[a-z0-9._-]+$

# as above but with upper case characters as well
PathAllowFilter ^[A-Za-z0-9._-]+$
```

# PathDenyFilter

## Name

PathDenyFilter –– FIXME FIXME

## Synopsis

**PathDenyFilter** [ `PathDenyFilter regular-expression`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.1.7 and later

## Description

Similar to PathAllowFilter, PathDenyFilter specifies a regular expression which must not match any uploaded pathnames. If the regex does match, a "Forbidden filename" error is returned to the client. This can be especially useful for forbidding .ftpaccess or .htaccess files.

## See also

## Examples

```
# We don't want .ftpaccess or .htaccess files to be uploaded
PathDenyFilter "(\\.ftpaccess)|(\\.htaccess)$"
```

# PersistentPasswd

## Name

PersistentPasswd –– FIXME FIXME

## Synopsis

**PersistentPasswd** [ `PersistentPasswd on|off`]

`Default`
Platform dependent
`Context`
server config
`Module`
mod_unixpw
`Compatibility`
1.1.5 and later

## Description

The PersistentPasswd directive controls how proftpd handles authentication, user/group lookups, and user/group to name mapping. If set to On, proftpd will attempt to open the system–wide /etc/passwd, /etc/group (and /etc/shadow, potentially) files itself, holding them open even during a chroot()ed login (note that /etc/shadow is never held open, for security reasons). On some platforms, you must turn this option on, as the libc functions are incapable of accessing these databases from inside of a chroot(). At configure–time, the configuration script will attempt to detect whether or not you need this support, and make it the default. However, such "guessing" may fail, and you will have to manually enable or disable the feature. If you cannot see user or group names when performing a directory listing inside an anonymous chrooted login, this indicates you must enable the directive. Use of the AuthUserFile or AuthGroupFile directives will force partial support for persistent user or group database files; regardless of PersistentPasswd's setting.

Note: NIS or NIS+ users will most likely want to disable this feature, regardless of proftpd's detected configuration defaults. Failure to disable this will make your NIS/NIS+ maps not work! On certain systems, you may also need to compile ProFTPD with the ––enable–autoshadow option in order to authenticate both users from NIS maps and local users.

## See also

## Examples

# PidFile

## Name

PidFile –– FIXME FIXME

## Synopsis

**PidFile** [ `PidFile filename`]

`Default`
none
`Context`
server config, <Global>
`Module`
mod_core
`Compatibility`
1.2.0rc2 and later

## Description

The PidFile directive sets the file to which the server records the process id of the daemon. The filename should be relative to the system root, ie /var/run/proftpd/pidfile. The PidFile is only used in standalone mode. It is often useful to be able to send the server a signal, so that it closes and then reopens its ErrorLog and TransferLog, and re–reads its configuration files. This is done by sending a SIGHUP (kill –1) signal to the process id of the master daemon listed in the PidFile.

## See also

## Examples

# Port

## Name

Port –– FIXME FIXME

## Synopsis

**Port** [ `Port port-number`]

`Default`
Port 21

`Context`
server config, <VirtualHost>

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The Port directive configures the TCP port which proftpd will listen on while running in standalone mode. It has no effect when used upon a server running in inetd mode (see ServerType). The directive can be used in conjunction with <VirtualHost> in order to run a virtual server on the same IP address as the master server, but listening on a different port.

For any server, either <VirtualHost> or server config, setting Port 0 effectively turns off that server.

## See also

## Examples

# PostgresInfo

## Name

PostgresInfo –– Postgres backend configuration (Deprecated)

## Synopsis

**PostgresInfo** [ `hostname`] [ [`sqluser`] [ `sqlpass`] ] [ `dbname`]

`Default`
none
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0rc2 and later

## Description

This directive is deprecated, please use SQLConnectInfo instead.

Configures the Posgresql database driver (the database may be remote). A connection isn't made until use of a SQL feature requires it, after which it may be held open for the lifetime of the FTP session depending on the directives in use.

## See also

## Examples

PostgresInfo myserver.example.com proftpd wibble ftpusers

# PostgresPort

## Name

PostgresPort –– Sets the port postgres is listening on

## Synopsis

**PostgresPort** [ `portnumber`]

`Default`
5432
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0rc2 and later

## Description

This directive is deprecated, please use SQLConnectInfo instead

Specifies which TCP/IP port to use for connecting. Default is 5432, or UNIX socket for localhost.

## See also

## Examples

PostgresPort 3306

# QuotaBlockName

## Name

QuotaBlockName –– FIXME FIXME

## Synopsis

**QuotaBlockName** [ `QuotaBlockName name`]

`Default`
byte
`Context`
server, <VirtualHost>, <Anonymous>
`Module`
mod_quota
`Compatibility`
at least 1.2.0 and later

## Description

The QuotaBlockName directive is used in conjunction with the QuotaBlockSize directive to control user output from the module. This directive specifies the name given to the values displayed (ie byte, kilobyte, kb etc etc). Example: QuotaBlockName kb

## See also

## Examples

# QuotaBlockSize

## Name

QuotaBlockSize –– FIXME FIXME

## Synopsis

**QuotaBlockSize** [ `QuotaBlockSize number of bytes`]

`Default`
None
`Context`
server, <VirtualHost>, <Anonymous>
`Module`
mod_quota
`Compatibility`
at least 1.2.0 and later

## Description

The QuotaBlockSize directive is used in conjuntion with the QuotaBlockName directive to control how the user output is handled. QuotaBlockSize specifies the factor by which the values in the user reports are divided before display. Example: QuotaBlockSize 1024

## See also

## Examples

# QuotaCalc

## Name

QuotaCalc –– FIXME FIXME

## Synopsis

**QuotaCalc** [ `QuotaCalc foo1 foo2 foo3`]

`Default`
None
`Context`
server, <VirtualHost>, <Anonymous>
`Module`
mod_quota
`Compatibility`
at least 1.2.0 and later

## Description

The QuotaCalc directive controls whether calculation is done on the fly. If the directive is set to 'on' and either there is no .quota file or the quota would go negative then calculation is done on the fly rather than at the end of the session.

## See also

## Examples

# QuotaExempt

## Name

QuotaExempt –– FIXME FIXME

## Synopsis

**QuotaExempt** [ `QuotaExempt uid, uid, uid`]

`Default`
None

`Context`
server, <VirtualHost>, <Anonymous>

`Module`
mod_quota

`Compatibility`
at least 1.2.0 and later

## Description

The QuotaExempt directive lists the UIDs which are not subject to quota controls, using UIDs rather than symbolic user names speeds up the loading and resolution process. Example: QuotaExempt 3000,3401,500

## See also

## Examples

# Quotas

## Name

Quotas –– FIXME FIXME

## Synopsis

**Quotas** [ `Quotas on|off`]

`Default`
none

`Context`
server, <VirtualHost>, <Anonymous>

`Module`
mod_quota

`Compatibility`
at least 1.2.0 and later

## Description

The Quotas directive enables or disables Quota support. Example: Quotas on

## See also

## Examples

# QuotaType

## Name

QuotaType –– FIXME FIXME

## Synopsis

**QuotaType** [ `QuotaType soft|hard`]

`Default`
soft
`Context`
server, <VirtualHost>, <Anonymous>
`Module`
mod_quota
`Compatibility`
at least 1.2.0 and later

## Description

The QuotaType directive defines what happens to files which break the quota limits as they are uploaded. Setting the type to hard ensures that the file which violates the quota is deleted. uploaded.

## See also

## Examples

# RateReadBPS

## Name

RateReadBPS –– FIXME FIXME

## Synopsis

**RateReadBPS** [ `RateReadBPS byte_per_sec-number`]

`Default`
0
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_xfer
`Compatibility`
1.2.0 and later

## Description

RateReadBPS sets the allowed byte per second download bandwidth in the given config context. Zero means no bandwidth limit. (See RateReadFreeBytes about limiting bandwidth only after some amount of downloaded bytes.) The usual place for this directive is in <VirtualHost> or <Directory> sections.

## See also

## Examples

# RateReadFreeBytes

## Name

RateReadFreeBytes –– FIXME FIXME

## Synopsis

**RateReadFreeBytes** [ `RateReadFreeBytes number of bytes`]

`Default`
0
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_xfer
`Compatibility`
1.2.0 and later

## Description

RateReadFreeBytes is the amount of bytes to be transferred without any bandwidth limits, so with that option you can give full bandwidth for small files while limiting big ones. (See RateReadHardBPS on further info about what happens after the free amount was transferred.)

## See also

## Examples

# RateReadHardBPS

## Name

RateReadHardBPS –– FIXME FIXME

## Synopsis

**RateReadHardBPS** [ `RateReadHardBPS on/off`]

Default
off
Context
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
Module
mod_xfer
Compatibility
1.2.0 and later

## Description

RateReadHardBPS forces the bandwidth to the given RateReadBPS value after the RateReadFreeBytes amount of file was transfered. This means that if the user have huge bandwidth and downloaded the "free" amount fast, HardBPS will stop the transfer until the average goes down to the given limit. If the amount of FreeBytes is high and the ReadBPS is low then the user may wait for extended periods of time until the transfer continues. :–)

## See also

## Examples

# RateWriteBPS

## Name

RateWriteBPS -- FIXME FIXME

## Synopsis

**RateWriteBPS** [ `RateWriteBPS byte_per_sec-number`]

`Default`
0
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_xfer
`Compatibility`
1.2.0 and later

## Description

RateWriteBPS sets the allowed byte per second upload bandwidth in the given config context. Zero means no bandwidth limit. (See RateWriteFreeBytes about limiting bandwidth only after some amount of uploaded bytes.) The usual place for this directive is in <VirtualHost> or <Directory> sections.

## See also

## Examples

# RateWriteFreeBytes

## Name

RateWriteFreeBytes –– FIXME FIXME

## Synopsis

**RateWriteFreeBytes** [ `RateWriteFreeBytes number of bytes`]

`Default`
0
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_xfer
`Compatibility`
1.2.0 and later

## Description

RateWriteFreeBytes is the amount of bytes to be transferred without any bandwidth limits, so with that option you can give full bandwidth for small files while limiting big ones. (See RateWriteHardBPS on further info about what happens after the free amount was transferred.)

## See also

## Examples

# RateWriteHardBPS

## Name

RateWriteHardBPS –– FIXME FIXME

## Synopsis

**RateWriteHardBPS** [ `RateWriteHardBPS on/off`]

`Default`
off
`Context`
server config, <VirtualHost>, <Anonymous>, <Directory>, <Global>
`Module`
mod_xfer
`Compatibility`
1.2.0 and later

## Description

RateWriteHardBPS forces the bandwidth to the given RateWriteBPS value after the RateWriteFreeBytes amount of file was transfered. This means that if the user have huge bandwidth and uploaded the "free" amount fast, HardBPS will stop the transfer until the average goes down to the given limit. If the amount of FreeBytes is high and the WriteBPS is low then the user may wait for extended periods of time until the transfer continues. :–) RateWriteHardBPS RatioFile (mod_ratio) Incomplete Ratios (mod_ratio) Incomplete RatioTempFile (mod_ratio) Incomplete

## See also

## Examples

# RatioFile

## Name

RatioFile –– Ratio directive

## Synopsis

**RatioFile** [ `RatioFile foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The RatioFile directive .... Example: RatioFile

## See also

## Examples

# Ratios

## Name

Ratios –– FIXME FIXME

## Synopsis

**Ratios** [ `Ratios foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The Ratios directive .... Example: Ratios

## See also

## Examples

# RatioTempFile

## Name

RatioTempFile –– Ratio directive

## Synopsis

**RatioTempFile** [ `RatioTempFile foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The RatioTempFile directive .... Example: RatioTempFile

## See also

## Examples

# RequireValidShell

## Name

RequireValidShell –– Allow connections based on /etc/shells

## Synopsis

**RequireValidShell** [ `RequireValidShell on|off`]

`Default`
RequireValidShell on
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The RequireValidShell directive configures the server, virtual host or anonymous login to allow or deny logins which do not have a shell binary listed in /etc/shells. By default, proftpd disallows logins if the user's default shell is not listed in /etc/shells. If /etc/shells cannot be found, all default shells are assumed to be valid.

## See also

## Examples

# RLimitCPU

## Name

RLimitCPU –– Configure the maximum CPU time in seconds used by a process

## Synopsis

**RLimitCPU** [ `RLimitCPU soft-limit|"max" [hard-limit|"max"]]`

`Default`
System defaults
`Context`
server config
`Module`
mod_core
`Compatibility`
1.2.1rc1 and later

## Description

RLimitCPU takes 1 or 2 parameters. The first parameter sets the soft resource limit for all proftpd processes. The optional second parameter sets the maximum resource limit. Either parameter can be a number, or max to indicate to the server that the limit should be set to the maximum allowed by the operating system configuration.

CPU resource limits are expressed in seconds per process.

## See also

RLimitMemory, RLimitOpenFiles

## Examples

# RLimitMemory

## Name

RLimitMemory –– Configure the maximum memory in bytes used by a process

## Synopsis

**RLimitMemory** [ `RLimitMemory soft-limit[units]|"max" [hard-limit[units]|"max"]]`

`Default`
: None

`Context`
: server config

`Module`
: mod_core

`Compatibility`
: 1.2.1rc1 and later

## Description

RLimitMemory takes 1 or 2 parameters. The first parameter sets the soft resource limit for all proftpd processes. The optional second parameter sets the maximum resource limit. Either parameter can be a number, or max to indicate to the server that the limit should be set to the maximum allowed by the operating system configuration.

Memory resource limits are expressed in bytes per process. An optional case–insensitive units specifier may follow the number of bytes given: G (Gigabytes), M (Megabytes), K (Kilobytes), or B (bytes). If the units specifier is used, the given number of bytes is multiplied by the appropriate factor.

## See also

RLimitCPU, RLimitMaxProcesses, RLimitOpenFiles

# RLimitOpenFiles

## Name

RLimitOpenFiles –– Configure the maximum number of open files used by a process

## Synopsis

**RLimitOpenFiles** [ `RLimitOpenFiles soft-limit|"max" [hard-limit|"max"]`]

`Default`
None
`Context`
server config
`Module`
mod_core
`Compatibility`
1.2.1rc1 and later

## Description

RLimitOpenFiles takes 1 or 2 parameters. The first parameter sets the soft resource limit for all proftpd processes. The optional second parameter sets the maximum resource limit. Either parameter can be a number, or max to indicate to the server that the limit should be set to the maximum allowed by the operating system configuration.

File resource limits are expressed in number of files per process.

## See also

RLimitCPU, RLimitMaxProcesses, RLimitMemory

# RootLogin

## Name

RootLogin –– Permit root user logins

## Synopsis

**RootLogin** [ `RootLogin on|off`]

`Default`
RootLogin off

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_auth

`Compatibility`
1.1.5 and later

## Description

Normally, proftpd disallows root logins under any circumstance. If a client attempts to login as root, using the correct password, a special security message is sent to syslog. When the RootLogin directive is turned On, the root user may authenticate just as any other user could (assuming no other access control measures deny access); however the root login security message is still sysloged. Obviously, extreme care should be taken when using this directive.

The use of RootLogin in the Anonymous context is only valid when the User / Group defined in the Anonymous block is set to 'root'

## See also

## Examples

# SaveRatios

## Name

SaveRatios –– FIXME FIXME

## Synopsis

**SaveRatios** [ `SaveRatios foo1 foo2 foo3`]

`Default`
None known

`Context`
<Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
mod_ratio

`Compatibility`
at least 1.2.0 and later

## Description

The SaveRatios directive .... Example: SaveRatios

## See also

## Examples

# ScoreboardPath

## Name

ScoreboardPath −− Sets the path to the scoreboard file

## Synopsis

**ScoreboardPath** [ `path`]

`Default`
ScoreboardPath /var/run

`Context`
server config

`Module`
mod_core

`Compatibility`
1.1.6 and later

## Description

The ScoreboardPath directive sets the directory where proftpd run−time scoreboard files (proftpd−*) are kept. These file(s) are necessary for MaxClients to work properly, as well as other utilities (such as ftpwho and ftpcount).

## See also

## Examples

# ServerAdmin

## Name

ServerAdmin –– Set the address for the server admin

## Synopsis

**ServerAdmin** [ `ServerAdmin "admin-email-address"`]

`Default`
ServerAdmin root@[ServerName]

`Context`
server config, <VirtualHost>

`Module`
mod_core

`Compatibility`
0.99.0pl10 and later

## Description

The ServerAdmin directive sets the email address of the administrator for the server or virtualhost. This address is displayed in magic cookie replacements (see DisplayLogin and DisplayFirstChdir).

## See also

## Examples

# ServerIdent

## Name

ServerIdent –– Set the message displayed on connect

## Synopsis

**ServerIdent** [ `ServerIdent off|on [identification string]`]

`Default`
ServerIdent ProFTPD [version] Server (server name) [hostname]

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_core

`Compatibility`
1.2.0pre2 and later

## Description

The ServerIdent directive sets the default message displayed when a new client connects. Setting this to off displays "[hostname] FTP server ready." If set to on, the directive can take an optional string argument, which will be displayed instead of the default text. Sites desiring to give out minimal information will probably want a setting like ServerIdent on "FTP Server ready.", which won't even reveal the hostname.

## See also

## Examples

ServerIdent on "Welcome to ftp.linux.co.uk"

# ServerName

## Name

ServerName –– Configure the name displayed to connecting users

## Synopsis

**ServerName** [ `ServerName "name"`]

`Default`
: ServerName "ProFTPD Server [version]"

`Context`
: server config, <VirtualHost>

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

The ServerName directive configures the string that will be displayed to a user connecting to the server (or virtual server if the directive is located in a <VirtualHost> block). See Also: <VirtualHost>

## See also

## Examples

# ServerType

## Name

ServerType –– Set the mode proftpd runs in

## Synopsis

**ServerType** [ `ServerType type–identifier`]

`Default`
: ServerType standalone

`Context`
: server config

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

The ServerType directive configures the server daemon's operating mode. The type–identifier can be one of two values: inetd The daemon will expect to be run from the inetd "super server." New connections are passed from inetd to proftpd and serviced immediately. standalone The daemon starts and begins listening to the configured port for incoming connections. New connections result in spawned child processes dedicated to servicing all requests from the newly connected client.

## See also

## Examples

# ShowDotFiles

## Name

ShowDotFiles –– Toggle display of 'dotfiles'

## Synopsis

**ShowDotFiles** [ `ShowDotFiles on|off`]

`Default`
ShowDotFiles Off
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_ls
`Compatibility`
0.99.0pl6 and later –– Deprecated

## Description

If set to on, files starting with a '.', except for the directories '.' and '..', will be displayed in directory listings. This directive has been deprecated in favor of LsDefaultOptions –– e.g., LsDefaultOptions "–A" –– and may be removed in future versions. See Also: LsDefaultOptions

## See also

## Examples

# ShowSymlinks

## Name

ShowSymlinks -- Toggle the display of symlinks

## Synopsis

**ShowSymlinks** [ `ShowSymlinks on|off`]

`Default`
: (versions 1.1.5 and beyond) ShowSymlinks On

`Context`
: server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
: mod_core

`Compatibility`

## Description

Compatibility: 0.99.0pl6 and later Symbolic links (if supported on the host OS and filesystem) can be either shown in directory listings (including the target of the link) or can be "hidden" (proftpd dereferences symlinks and reports the target's permissions and ownership). The default behavior is to show all symbolic links when normal users are logged in, and hide them for anonymous sessions. If a symbolic link cannot be dereferenced for any reason (permissions, target does not exist, etc) and ShowSymlinks is off, proftpd displays the link as a directory entry of type 'l' (link) with the ownership and permissions of the actual link. Under ProFTPD versions 1.1.5 and higher, the default behavior in regard to ShowSymlinks has been changed so that symbolic links are always displayed as such (in all cases), unless ShowSymlinks off is explicitly set.

## See also

## Examples

# SocketBindTight

## Name

SocketBindTight −− Controls how TCP/IP sockets are created

## Synopsis

**SocketBindTight** [ `SocketBindTight on|off`]

`Default`
SocketBindTight off
`Context`
server config
`Module`
mod_core
`Compatibility`
0.99.0pl6 and later

## Description

The SocketBindTight directive controls how proftpd creates and binds its initial tcp listen sockets in standalone mode (see ServerType). The directive has no effect upon servers running in inetd mode, because listen sockets are not needed or created. When SocketBindTight is set to off (the default), a single listening socket is created for each port that the server must listen on, regardless of the number of IP addresses being used by <VirtualHost> configurations. This has the benefit of typically requiring a relatively small number of file descriptors for the master daemon process, even if a large number of virtual servers are configured. If SocketBindTight is set to on, a listen socket is created and bound to a specific IP address for the master server and all configured virtual servers. This allows for situations where an administrator may wish to have a particular port be used by both proftpd (on one IP address) and another daemon (on a different IP address). The drawback is that considerably more file descriptors will be required if a large number of virtual servers must be supported. Example: Two servers have been configured (one master and one virtual), with the IP addresses 10.0.0.1 and 10.0.0.2, respectively. The 10.0.0.1 server runs on port 21, while 10.0.0.2 runs on port 2001. SocketBindTight off #default # proftpd creates two sockets, both bound to ALL available addresses. # one socket listens on port 21, the other on 2001. Because each socket is # bound to all available addresses, no other daemon or user process will be # allowed to bind to ports 21 or 2001. ... SocketBindTight on # proftpd creates two sockets again, however one is bound to 10.0.0.1, port 21 # and the other to 10.0.0.2, port 2001. Because these sockets are "tightly" # bound to IP addresses, port 21 can be reused on any address OTHER than # 10.0.0.1, and visa−versa with 10.0.0.2, port 2001. One side−effect of setting SocketBindTight to on is that connections to non−bound addresses will result in a "connection refused" message rather than the typical "500 Sorry, no server available to handle request on xxx.xxx.xxx.xxx.", due to the fact that no listen socket has been bound to the particular address/port pair. This may or may not be aesthetically desirable, depending on your circumstances.

## See also

## Examples

# SQLAuthenticate

## Name

SQLAuthenticate –– Specify authentication methods and what to authenticate

## Synopsis

**SQLAuthenticate** {on | off}

or

**SQLAuthenticate** [ users [*] ] [ group [*] ] [ userset [fast] ] [ groupset [fast] ]

`Default`
: SQLAuthenticate `on`

`Context`
: server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
: mod_sql

`Compatibility`
: 1.2.5rc1 and later

## Description

The SQLAuthenticate directive controls the behavior of mod_sql regarding the authentication process. SQLAuthenticate can provide fine grained control over authentication of logins and file access for both users and groups. Using this directive, mod_sql can be configured to be the authoritative authentication mechanism – in that case, mod_sql provides authentication and all other authentication mechanisms will be bypassed.

The syntax for SQLAuthenticate can take one of two possible formats. The simplest syntax is a simple `on | off` format:

*on*
: mod_sql will perform login authentication and will also control file access using both user ID and group ID. This is equivalent to the following alternative syntax:

```
SQLAuthenticate users groups userset groupset
```

*off*
: mod_sql will not perform user or group lookups nor will it control file access or functionality.

A more complex syntax is provided to provide finer control of the behavior of mod_sql. Two features in particular may be controlled via this syntax:

- Authorititative lookups and authentication

-

File access or functionality control based on UID or GID

The following command options are used to control these features. Note that each of these options may be listed in any order.

*users[*]*
: If this option is present, user lookups will take place. Appending an asterisk to `users` will cause mod_sql to become authoritiative for user lookups. All other user authentication methods will be ignored. If this option is not included, mod_sql will not perform any user lookups.

*groups[*]*
: If this option is present, group lookups will take place. Appending an asterisk to `groups` will cause mod_sql to become authoritiative for group lookups. All other authentication methods will be ignored. If this option is not included, mod_sql will not perform any group lookups.

*userset[fast]*
: If this option is present, mod_sql will control file access or functionality by processing the (get|set|end)pwent calls. These calls are used to determine file access rights based on username. This option has no effect if the `user[*]` option is not present.

  If mod_sql is used to authenticate a significant number of users, the (set|get|end)pwent calls can become expensive. The number of queries will be n+1, where n is the number of users to be looked up. On a large system, this can significantly slow logins. Using the `usersetfast` option will cause a single query to be performed to lookup all users, speeding up the login process. The drawback to this option is that memory utilization will be increased.

*groupset[fast]*
: If this option is present, mod_sql will control file access or functionality by processing the (get|set|end)grent calls. These calls are used to determine file access rights based on groupname. This option has no effect if the `group[*]` option is not present.

  If mod_sql is used to authenticate a significant number of groups, the (set|get|end)grent calls can become expensive. The number of queries will be n+1, where n is the number of groups to be looked up. On a large system, this can significantly slow logins. Using the `groupsetfast` option will cause a single query to be performed to lookup all groups, speeding up the login process. The drawback to this option is that memory utilization will be increased.

Turning off (not including) userset or groupset affects the functionality of mod_sql. Not allowing these lookups may remove the ability to control access or control functionality by group membership, depending on your other auth handlers and the data available to them. At the same time, choosing not to do these lookups may dramatically speed login for many large sites.

The 'fast' suffix is not appropriate for every site. Normally, mod_sql will retrieve a list of users and groups, and get information from the database on a per−user or per−group basis. This is query intensive −− it requires (n+1) queries, where n is the number of users or groups to lookup. By choosing 'fast' lookups, mod_sql will make a single SELECT query to get information from the database.

In exchange for the radical reduction in the number of queries, the single query will increase the memory consumption of the process −− all group or user information will be read at once rather than in discrete chunks.

**Note:**If the `groupset` option is specified, mod_sql requires that the SQL group table contain only a single record for each group. All members of a group must be specified in the single record. Make sure that the group table is created with a sufficent column size for group members − for example, a MySQL group table should use type `TEXT` for the group members column, providing 65535 characters for listing all of the group members in a comma−separated list.

# See also

SQLUserTable , SQLGroupTable , SQLUserInfo , SQLGroupInfo

# Examples

If user and group lookups are desired, but other means will be used to perform file access control, and the user/group lookups are not to be authoritatuve, the following directive syntax is appropriate. This is not a particuarly interesting configuration.

```
SQLAuthenticate users groups
```

A more interesting configuration for mod_sql is shown below. In this configuration, mod_sql is authoritative for both users and groups, and also performs access control based on both user name and group membership. Utilizing a configuration such as this removes the need to provide a shell account for users on the server, while still providing "non−anonymous" ftp access with access control. The "fast" option is also used to speed up logins, at the expense of increased memory utilization.

```
SQLAuthenticate users* groups* usersetfast groupsetfast
```

# SQLAuthoritative

## Name

SQLAuthoritative -- FIXFIXFIX

## Synopsis

**SQLAuthoritative** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLAuthTypes

## Name

SQLAuthTypes −− FIXME FIXME

## Synopsis

**SQLAuthTypes** [ [OpenSSL]] [ [Crypt]] [ [Backend]] [ [Plaintext]] [ [Empty]]

Default
: none

Context
: server config, <Global>, <VirtualHost>

Module
: mod_sql

Compatibility
: 1.2.0 and later

## Description

This directive deprecates 'SQLEmptyPasswords', 'SQLScrambledPasswords', 'SQLSSLHashedPasswords', 'SQLPlaintextPasswords', and 'SQLEncryptedPasswords'. Specifies the allowed authentication types and their check order. YOU MUST SPECIFY AT LEAST ONE AUTHENTICATION METHOD. For example: SQLAuthTypes Crypt Empty means check whether the password in the database matches in UNIX crypt() format; if that fails, check to see if the password in the database is empty (matching ANY given password); if that fails, mod_sql refuses to authenticate the user. Current Types Plaintext: allows passwords in the database to be in plaintext OpenSSL: allows passwords in the database to be of the form '{digestname}hashedvalue'. This check is only available if you define 'HAVE_OPENSSL' when you compile proftd and you link with the OpenSSL 'crypto' library. Crypt: allows passwords in the database to be in UNIX crypt() form Backend: a database−specific backend check function. Not all backends support this. Specifically, the MySQL backend uses this type to authenticate MySQL 'PASSWORD()' encrypted passwords. The Postgres backend does nothing. Empty: allows empty passwords in the database, which match against ANYTHING the user types in. The database field must be a truly empty string −− that is, NULL values are never accepted. BE VERY CAREFUL WITH THIS AUTHTYPE.

# SQLConnectInfo

## Name

SQLConnectInfo −− FIXME FIXME

## Synopsis

**SQLConnectInfo** [ `connection−info`] [ `[username]`] [ `[password]`]

`Default`
: none

`Context`
: server config, <Global>, <VirtualHost>

`Module`
: mod_sql

`Compatibility`
: 1.2.0 and later

## Description

This directive deprecates 'MySQLInfo', 'PostgresInfo', and 'PostgresPort'. Specifies connection information. Connection−info specifies the database, host, port, and other backend−specific information. username and password specify the username and password to connect as, respectively. Both default to NULL, which the backend will treat in some backend−specific manner. If you specify a password, you MUST specify a username. Any given backend has the opportunity (but not the responsibility) to check for syntax errors in the connection−info field at proftpd startup, but you shouldn't expect semantic errors (i.e., can't connect to the database) to be caught until mod_sql attempts to connect for a given host. The MySQL and Postgres backends connection−info is expected to be of the form: database[@hostname][:port] hostname will default to a backend−specific hostname (which happens to be 'localhost' for both the MySQL and Postgres backends), and port will default to a backend−specific default port (3306 for the MySQL backend, 5432 for the Postgres backend). Examples: SQLConnectInfo ftpusers@foo.com means "Try connecting to the database 'ftpuser' via the default port at 'foo.com'. Use a NULL username and a NULL password." SQLConnectInfo ftpusers:3000 admin means "Try connecting to the database 'ftpuser' via port 3000 at 'localhost'. Use the username 'admin' and a NULL password." SQLConnectInfo ftpusers@foo.com:3000 admin mypassword means "Try connecting to the database 'ftpuser' via port 3000 at 'foo.com'. Use the username 'admin' and the password 'mypassword'" Backends may require different information in the connection−info field; check your backend module for specifics.

# SQLDefaultGID

## Name

SQLDefaultGID –– FIXME FIXME

## Synopsis

**SQLDefaultGID** [ `defaultgid`]

`Default`
65533

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Sets the default GID for users. Must be greater than SQLMinID.

# SQLDefaultHomedir

## Name

SQLDefaultHomedir –– FIXFIXFIX

## Synopsis

**SQLDefaultHomedir** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLDefaultUID

## Name

SQLDefaultUID –– FIXME FIXME

## Synopsis

**SQLDefaultUID** [ `defaultuid`]

`Default`
65533

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Sets the default UID for users. Must be greater than SQLMinID.

# SQLDoAuth

## Name

SQLDoAuth –– FIXME FIXME

## Synopsis

**SQLDoAuth** [ `on` | `off`]

`Default`
on

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Activates SQL authentication. This overrides all other directives –– SQLDoGroupAuth and SQLAuthoritative are ineffectual if SQLDoAuth is off.

# SQLDoGroupAuth

## Name

SQLDoGroupAuth –– FIXME FIXME

## Synopsis

**SQLDoGroupAuth** [ `on`|`off`]

`Default`
on
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0 and later

## Description

This directive causes mod_sql to pretend it has no group information. It necessarily breaks ALL CONFIG FILES up to 1.2.0rc2, since mod_sql now assumes that group information is available UNLESS this directive is set to OFF. This DOESN'T override SQLAuthoritative –– if SQLAuthoritative is set to 'On' but SQLDoGroupAuth is set to 'Off', all group–related queries will fail without giving other modules the opportunity to handle them. Prior to 1.2.0, there was no way to provide group information from the database. This caused a few bugs, and reduced the functionality of this module.

# SQLEmptyPasswords

## Name

SQLEmptyPasswords −− Allow zero length passwords (DEPRECATED)

## Synopsis

**SQLEmptyPasswords** [ `on`|`off`]

`Default`
off
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0rc2 and later

## Description

This directive is deprecated, please use SQLAuthTypes instead

Specifies whether an empty (non−NULL but zero−length) password is acceped from the database. Default is no, and truly NULL passwords are never accepted. If the retrieved password is empty then whatever password the user typed is accepted as valid, but the module logs a warning at debug level 4.

## See also

## Examples

SQLEmptyPasswords on

# SQLEncryptedPasswords

## Name

SQLEncryptedPasswords –– Assume SQL passwords are encrypted (DEPRECATED)

## Synopsis

**SQLEncryptedPasswords** [ `on` | `off`]

`Default`
on
`Context`
server config
`Module`
mod_sql
`Compatibility`
1.2.0rc2 and later

## Description

This directive is deprecated, please SQLAuthTypes instead

Specifies whether the password in the database may be in UNIX crypt() format. Default is true, with this being the only check done. A tool for generating crypted password text may be found at ftp://ftp.linpeople.org/pub/People/lilo/source/makepasswd–1.07.tar.gz

## See also

## Examples

SQLEncryptedPasswords on

# SQLGidField

## Name

SQLGidField –– FIXFIXFIX

## Synopsis

**SQLGidField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLGroupGIDField

## Name

SQLGroupGIDField –– FIXFIXFIX

## Synopsis

**SQLGroupGIDField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLGroupInfo

## Name

SQLGroupInfo -- FIXFIXFIX

## Synopsis

**SQLGroupInfo** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLGroupMembersField

## Name

SQLGroupMembersField –– FIXME FIXME

## Synopsis

**SQLGroupMembersField** [ `fieldname`]

`Default`
members
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0 and later

## Description

Specifies the field in the group table that holds the group's member list.

# SQLGroupnameField

## Name

SQLGroupnameField –– FIXME FIXME

## Synopsis

**SQLGroupnameField** [ `Syntax: fieldname`]

`Default`
groupname

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Specifies the field in the group table that holds the group name.

# SQLGroupTable

## Name

SQLGroupTable –– FIXME FIXME

## Synopsis

**SQLGroupTable** [ `tablename`]

`Default`
groups
`Context`
server config, <Global>, <VirtualHost>
`Module`
mod_sql
`Compatibility`
1.2.0 and later

## Description

Specifies the name of the table that holds group information.

# SQLGroupWhereClause

## Name

SQLGroupWhereClause –– FIXFIXFIX

## Synopsis

**SQLGroupWhereClause** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLHomedir

## Name

SQLHomedir –– FIXFIXFIX

## Synopsis

**SQLHomedir** [ "name" limit|regex|ip value]

Default
: FIXFIXFIX

Context
: server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

Module
: mod_sql

Compatibility
: 1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLHomedirField

## Name

SQLHomedirField –– FIXFIXFIX

## Synopsis

**SQLHomedirField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLHomedirOnDemand

## Name

SQLHomedirOnDemand –– FIXME FIXME

## Synopsis

**SQLHomedirOnDemand** [ `on`|`off`]

`Default`
off

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Specifies whether to automatically create a user's home directory if it doesn't exist at login.

# SQLLog

## Name

SQLLog -- FIXFIXFIX

## Synopsis

**SQLLog** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLLogDirs

## Name

SQLLogDirs –– FIXFIXFIX

## Synopsis

**SQLLogDirs** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLLogHits

## Name

SQLLogHits –– FIXFIXFIX

## Synopsis

**SQLLogHits** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLLogHosts

## Name

SQLLogHosts –– FIXFIXFIX

## Synopsis

**SQLLogHosts** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLLoginCountField

## Name

SQLLoginCountField –– FIXFIXFIX

## Synopsis

**SQLLoginCountField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLLogStats

## Name

SQLLogStats –– FIXFIXFIX

## Synopsis

**SQLLogStats** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLMinID

## Name

SQLMinID –– FIXME FIXME

## Synopsis

**SQLMinID** [ `minimumid`]

`Default`
999

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

SQLMinID is checked whenever retrieving a user's GID or UID. If the retrieved values for GID or UID are less than the value of SQLMinID, they are reported as the values of, respectively, 'SQLDefaultGID' and 'SQLDefaultUID'.

# SQLMinUserGID

## Name

SQLMinUserGID -- FIXFIXFIX

## Synopsis

**SQLMinUserGID** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLMinUserUID

## Name

SQLMinUserUID -- FIXFIXFIX

## Synopsis

**SQLMinUserUID** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLNamedQuery

## Name

SQLNamedQuery –– FIXFIXFIX

## Synopsis

**SQLNamedQuery** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLPasswordField

## Name

SQLPasswordField –– FIXFIXFIX

## Synopsis

**SQLPasswordField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLProcessGrEnt

## Name

SQLProcessGrEnt -- FIXFIXFIX

## Synopsis

**SQLProcessGrEnt** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLProcessPwEnt

## Name

SQLProcessPwEnt –– FIXFIXFIX

## Synopsis

**SQLProcessPwEnt** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLRatios

## Name

SQLRatios -- FIXFIXFIX

## Synopsis

**SQLRatios** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLRatioStats

## Name

SQLRatioStats –– FIXFIXFIX

## Synopsis

**SQLRatioStats** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLScrambledPasswords

## Name

SQLScrambledPasswords –– FIXME FIXME

## Synopsis

**SQLScrambledPasswords** [ `on` | `off`]

`Default`
off

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

This directive is DEPRECATED. Please use SQLAuthTypes instead. Specifies whether to accept passwords in a backend specific format. For the MySQL backend, this means 'PASSWORD()' scrambled passwords. For the Postgres backend, this check does nothing.

# SQLShellField

## Name

SQLShellField –– FIXME FIXME

## Synopsis

**SQLShellField** [ `fieldname`]

`Default`
shell

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

Specifies the field in the user table that holds the user's shell. If this field doesn't exist or the result of the query is NULL, the shell is reported as "".

# SQLShowInfo

## Name

SQLShowInfo –– FIXFIXFIX

## Synopsis

**SQLShowInfo** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLSSLHashedPasswords

## Name

SQLSSLHashedPasswords –– FIXME FIXME

## Synopsis

**SQLSSLHashedPasswords** [ `on` | `off`]

`Default`
: off

`Context`
: server config, <Global>, <VirtualHost>

`Module`
: mod_sql

`Compatibility`
: 1.2.0 and later

## Description

This directive is DEPRECATED. Please use SQLAuthTypes instead. Specifies whether to accept passwords of the form {digestname}hashedpassword from the database. This directive is only available if you define 'HAVE_OPENSSL' when you compile proftd and you link with the OpenSSL 'crypto' library. As an example, any of the following password entries in the database would match if the user typed the password 'testpassword': {SHA}IoFZRnP0iujh/70lps6DjKPgwkk= {SHA1}i7YRj4/Wk1rQh2o740pxfTJwj/0= {MD2}nS6iguewvAdrCnOMyQjB1w== {MD4}5wsGtJCkyXBzDJoVsQKjSg== {MD5}4WsquNEjFL9O+9YgOQbqbA==

# SQLUidField

## Name

SQLUidField –– FIXFIXFIX

## Synopsis

**SQLUidField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLUserInfo

## Name

SQLUserInfo –– FIXFIXFIX

## Synopsis

**SQLUserInfo** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX

`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess

`Module`
mod_sql

`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLUsernameField

## Name

SQLUsernameField –– FIXFIXFIX

## Synopsis

**SQLUsernameField** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLUserTable

## Name

SQLUserTable –– FIXFIXFIX

## Synopsis

**SQLUserTable** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLUserWhereClause

## Name

SQLUserWhereClause -- FIXFIXFIX

## Synopsis

**SQLUserWhereClause** [ `"name" limit|regex|ip value`]

`Default`
FIXFIXFIX
`Context`
server config, <Global>, <VirtualHost>, <Anonymous>, <Limit>, .ftpaccess
`Module`
mod_sql
`Compatibility`
1.2.5rc1 and later

## Description

FIX FIX FIX

## See also

## Examples

FIXFIXFIX

FIXFIX

# SQLWhereClause

## Name

SQLWhereClause –– FIXME FIXME

## Synopsis

**SQLWhereClause** [ `whereclause`]

`Default`
none

`Context`
server config, <Global>, <VirtualHost>

`Module`
mod_sql

`Compatibility`
1.2.0 and later

## Description

This directive deprecates 'SQLKey' and 'SQLKeyField'. Specifies a where clause that is added to every user query (this has no effect on group queries). The where clause *must* contain all relevant punctuation, and *must not* contain a leading 'and'. As an example of switching from the old–style 'SQLKey' and 'SQLKeyField' directives, if you had: SQLKey true SQLKeyfield LoginAllowed You would now use: SQLWhereClause "LoginAllowed = 'true'" This would be appended to every user–related query as the string " and (LoginAllowed = 'true')"

# SyslogFacility

## Name

SyslogFacility –– Set the facility level used for logging

## Synopsis

**SyslogFacility** [ `SyslogFacility facility-level`]

`Default`
None
`Context`
server config
`Module`
mod_core
`Compatibility`
1.1.6 and later

## Description

Proftpd logs its activity via the Unix syslog mechanism, which allows for several different general classifications of logging messages, known as "facilities." Normally, all authentication related messages are logged with the AUTHPRIV (or AUTH) facility [intended to be secure, and never seen by unwanted eyes], while normal operational messages are logged with the DAEMON facility. The SyslogFacility directive allows ALL logging messages to be directed to a different facility than the default. When this directive is used, ALL logging is done with the specified facility, both authentication (secure) and otherwise. The facility–level argument must be one of the following: AUTH (or AUTHPRIV), CRON, DAEMON, KERN, LPR, MAIL, NEWS, USER, UUCP, LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6 or LOCAL7. See Also: SystemLog

## See also

## Examples

# SyslogLevel

## Name

SyslogLevel –– Set the verbosity level of system logging

## Synopsis

**SyslogLevel** [ `SyslogLevel emerg|alert|crit|error|warn|notice|info|debug`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
1.2.0rc2+cvs and later

## Description

SyslogLevel adjusts the verbosity of the messages recorded in the error logs. The following levels are available, in order of decreasing significance: Level Description emerg Emergencies – system is unusable. alert Action must be taken immediately. crit Critical Conditions. error Error conditions. warn Warning conditions. notice Normal but significant condition. info Informational. debug Debug–level messages When a particular level is specified, messages from all other levels of higher significance will be reported as well. E.g., when SyslogLevel info is specified, then messages with log levels of notice and warn will also be posted. Using a level of at least crit is recommended.

## See also

## Examples

# SystemLog

## Name

SystemLog –– Redirect syslogging to a file

## Synopsis

**SystemLog** [ `SystemLog filename`|`NONE`]

`Default`
: None

`Context`
: server config

`Module`
: mod_log

`Compatibility`
: 1.1.6pl1 and later

## Description

The SystemLog directive disables proftpd's use of the syslog mechanism and instead redirects all logging output to the specified filename. The filename argument should contain an absolute path, and should not be to a file in a nonexistent directory, in a world−writeable directory, or be a symbolic link (unless AllowLogSymlinks is set to on). Use of this directive overrides any facility set by the SyslogFacility directive. Additionally, the special keyword NONE can be used which disables all syslog style logging for the entire configuration.

## See also

AllowLogSymlinks

## Examples

# TCPAccessFiles

## Name

TCPAccessFiles –– Sets the access files to use

## Synopsis

**TCPAccessFiles** [ `allow–filename deny–filename`]

`Default`
none
`Context`
server config, <VirtualHost>, <Global>, <Anonymous>
`Module`
mod_wrap
`Compatibility`
1.2.1 and later

## Description

TCPAccessFiles specifies two files, an allow and a deny file, each of which contain the IP addresses, networks or name–based masks to be allowed or denied connections to the server. The files have the same format as the standard tcpwrappers hosts.allow/deny files.

Both file names are required. Also, the paths to both files must be the full path, with two exceptions: if the path starts with ~/, the check of that path will be delayed until a user requests a connection, at which time the path will be resolved to that user's home directory; or if the path starts with ~user/, where user is some system user. In this latter case, mod_wrap will attempt to resolve and verify the given user's home directory on start–up.

The service name for which mod_wrap will look in the indicated access files is proftpd by default; this can be configured via the TCPServiceName directive. There is a built–in precedence to the TCPAccessFiles, TCPGroupAccessFiles, and TCPUserAccessFiles directives, if all are used. mod_wrap will look for applicable TCPUserAccessFiles for the connecting user first. If no applicable TCPUserAccessFiles is found, mod_wrap will search for TCPGroupAccessFiles which pertain to the connecting user. If not found, mod_wrap will then look for the server–wide TCPAccessFiles directive. This allows for access control to be set on a per–server basis, and allow for per–user or per–group access control to be handled without interfering with the server access rules.

## See also

TCPGroupAccessFiles, TCPServiceName, TCPUserAccessFiles

## Examples

# server−wide access files TCPAccessFiles /etc/ftpd.allow /etc/ftpd.deny # per−user access files, which are to be found in the user's home directory TCPAccessFiles ~/my.allow ~/my.deny

# TCPAccessSyslogLevels

## Name

TCPAccessSyslogLevels −− Sets the logging levels for mod_wrap

## Synopsis

**TCPAccessSyslogLevels** [ `allow-level deny-level`]

`Default`
TCPAccessSyslogLevels info warn
`Context`
server config, <VirtualHost>, <Global>, <Anonymous>
`Module`
mod_wrap
`Compatibility`
1.2.1 and later

## Description

ProFTPD can log when a connection is allowed, or denied, as the result of rules in the files specified in TCPAccessFiles, to the Unix syslog mechanism. A discussion on the syslog levels which can be used is given in the SyslogLevel directive.

The allow−level parameter sets the syslog level at which allowed connections are logged; the deny−level parameter sets the syslog level for denied connections.

## See also

SyslogLevel

## Examples

TCPAccessSyslogLevels debug warn

# tcpBackLog

## Name

tcpBackLog –– Control the tcp backlog in standalone mode

## Synopsis

**tcpBackLog** [ `tcpBackLog backlog-size`]

`Default`
: tcpBackLog 5

`Context`
: server config

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

The tcpBackLog directive controls the tcp "backlog queue" when listening for connections in standalone mode (see ServerType). It has no affect upon servers in inetd mode. When a tcp connection is established by the tcp/ip stack inside the kernel, there is a short period of time between the actual establishment of the connection and the acceptance of the connection by a user–space program. The duration of this latency period is widely variable, and can depend upon several factors (hardware, system load, etc). During this period tcp connections cannot be accepted, as the port that was previously "listening" has become filled with the new connection. Under heavy connection load this can result in occasional (or even frequent!) "connection refused" messages returned to the incoming client, even when there is a service available to handle requests. To eliminate this problem, most modern tcp/ip stacks implement a "backlog queue" which is simply a pre–allocation of resources necessary to handle backlog–size connections during the latency period. The larger the backlog queue, the more connections can be established in a very short time period. The trade–off, of course, is kernel memory and/or other kernel resources. Generally it is not necessary to use a tcpBackLog directive, unless you intend to service a large number of virtual hosts (see <VirtualHost>), or have a consistently heavy system load. If you begin to notice or hear of "connection refused" messages from remote clients, try setting a slightly higher value to this directive.

## See also

## Examples

# TCPGroupAccessFiles

## Name

TCPGroupAccessFiles –– Sets the access files to use

## Synopsis

**TCPGroupAccessFiles** [ `group-expression allow-filename deny-filename`]

`Default`
none
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_wrap
`Compatibility`
1.2.1 and later

## Description

TCPGroupAccessFiles allows for access control files, the same types of files required by TCPAccessFiles, to be applied to select groups. The given group–expression is a logical AND expression, which means that the connecting user must be a member of all the groups listed for this directive to apply. Group names may be negated with a ! prefix.

The rules for the filename paths are the same as for TCPAccessFiles settings.

## See also

TCPAccessFiles, TCPUserAccessFiles

## Examples

# every member of group wheel must connect from restricted locations TCPGroupAccessFiles wheel /etc/ftpd–strict.allow /etc/ftpd–strict.deny # everyone else gets the standard access rules TCPGroupAccessFiles !wheel /etc/hosts.allow /etc/hosts.deny

# tcpNoDelay

## Name

tcpNoDelay –– Control the use of TCP_NODELAY

## Synopsis

**tcpNoDelay** [ `tcpNoDelay on|off`]

`Default`
tcpNoDelay on

`Context`
server config, <VirtualHost>, <Global>

`Module`
mod_core

`Compatibility`
1.2.0pre3a and later

## Description

The tcpNoDelay directive controls the use of the TCP_NODELAY socket option (which disables the Nagle algorithm). ProFTPd uses TCP_NODELAY by default, which usually is a benefit but this can occasionally lead to problems with some clients, so tcpNoDelay is provided as a way to disable this option. You will not normally need to use this directive but if you have clients reporting unusually slow connections, try setting this to off.

## See also

## Examples

# tcpReceiveWindow

## Name

tcpReceiveWindow −− Set the size of the tcp receive window

## Synopsis

**tcpReceiveWindow** [ `tcpReceiveWindow window−size`]

`Default`
tcpReceiveWindow 8192

`Context`
server config, <VirtualHost>

`Module`
mod_core

`Compatibility`
0.99.0 and later

## Description

The tcpReceiveWindow directive configures the size (in octets) of all data connections' tcp receive windows. It is only used when receiving a file from a client over the data connection. Typically, a given tcp/ip implementation will use a relatively small receive window size (the number of octets that can be received at the tcp layer before a "turnaround" acknowledgement is required). When transferring a large amount of data over fast digital transmission lines which have a relatively high latency, a small receive window can dramatically affect perceived throughput because of the necessity to completely stop the transfer occasionally in order to wait for the remote endpoint to receive the acknowledgement and continue transmission. For example, on a T1 line (assuming full 1.544Mbps endpoint−to−endpoint throughput) with 100 ms latency, a 4k receive buffer will very dramatically reduce the perceived throughput. The default value of 8192 octets (8k) should be reasonable in common network configurations. Additionally, proftpd allocates its internal buffers to match the receive/send window sizes; in order to maximize the reception/transmission performance (reducing the number of times data must be transfered from proftpd to the kernel tcp/ip stack). The tradeoff, of course, is memory; both kernel− and user−space. If running proftpd on a memory tight host (and on a low−bandwidth connection), it might be advisable to decrease both the tcpReceiveWindow and tcpSendWindow sizes.

## See also

## Examples

# tcpSendWindow

## Name

tcpSendWindow –– Set the size of the tcp send window

## Synopsis

**tcpSendWindow** [ `tcpSendWindow window-size`]

`Default`
tcpSendWindow 8192
`Context`
server config, <VirtualHost>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The tcpSendWindow directive configures the size (in octets) of all data connections' tcp send windows. It is only used when sending a file from the server to a client on the data connection. For a detailed description of receive/send window sizes see tcpReceiveWindow.

## See also

## Examples

# TCPServiceName

## Name

TCPServiceName –– Configures the name proftpd will use with mod_wrap

## Synopsis

**TCPServiceName** [ `name`]

`Default`
TCPServiceName proftpd
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_wrap
`Compatibility`
1.2.1 and later

## Description

TCPServiceName is used to configure the name of the service under which mod_wrap will check the allow/deny files. By default, this is the name of the program started, i.e. "proftpd". However, some administrators may want to use a different, more generic service name, such as "ftpd"; use this directive for such needs.

## See also

# TCPUserAccessFiles

## Name

TCPUserAccessFiles –– Sets the access files to use

## Synopsis

**TCPUserAccessFiles** [ `user−expression allow−filename deny−filename`]

`Default`
none
`Context`
server config, <VirtualHost>, <Global>
`Module`
mod_wrap
`Compatibility`
1.2.1 and later

## Description

TCPUserAccessFiles allows for access control files, the same types of files required by TCPAccessFiles, to be applied to select users. The given user−expression is a logical AND expression. Listing multiple users in a user−expression does not make much sense; however, this type of AND evaluation allows for expressions such as "everyone except this user" with the use of the ! negation prefix.

The rules for the filename paths are the same as for TCPAccessFiles settings.

## See also

TCPAccessFiles, TCPGroupAccessFiles

## Examples

# user admin might be allowed to connect from anywhere TCPUserAccessFiles admin /etc/ftpd−anywhere.allow /etc/ftpd−anywhere.deny # while every other user has to connect from LAN addresses TCPUserAccessFiles !admin /etc/ftpd−lan.allow /etc/ftpd−lan.deny

# TimeoutIdle

## Name

TimeoutIdle –– Sets the idle connection timeout

## Synopsis

**TimeoutIdle** [ `TimeoutIdle seconds`]

`Default`
TimeoutIdle 600
`Context`
server config
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The TimeoutIdle directive configures the maximum number of seconds that proftpd will allow clients to stay connected without receiving any data on either the control or data connection. If data is received on either connection, the idle timer is reset. Setting TimeoutIdle to 0 disables the idle timer completely (clients can stay connected for ever, without sending data). This is generally a bad idea as a "hung" tcp connection which is never properly disconnected (the remote network may have become disconnected from the Internet, etc) will cause a child server to never exit (at least not for a considerable period of time) until manually killed See Also: TimeoutLogin, TimeoutNoTransfer

## See also

## Examples

# TimeoutLogin

## Name

TimeoutLogin –– Sets the login timeout

## Synopsis

**TimeoutLogin** [ `TimeoutLogin seconds`]

`Default`
TimeoutLogin 300
`Context`
server config
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The TimeoutLogin directive configures the maximum number of seconds a client is allowed to spend authenticating. The login timer is not reset when a client transmits data, and is only removed once a client has transmitted an acceptable USER/PASS command combination. See Also: TimeoutIdle, TimeoutNoTransfer

## See also

## Examples

# TimeoutNoTransfer

## Name

TimeoutNoTransfer –– Sets the connection without transfer timeout

## Synopsis

**TimeoutNoTransfer** [ `TimeoutNoTransfer seconds`]

`Default`
TimeoutNoTransfer 300
`Context`
server config
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The TimeoutNoTransfer directive configures the maximum number of seconds a client is allowed to spend connected, after authentication, without issuing a command which results in creating an active or passive data connection (i.e. sending/receiving a file, or receiving a directory listing). See Also: TimeoutIdle, TimeoutLogin

## See also

## Examples

# TimeoutStalled

## Name

TimeoutStalled –– Sets the timeout on stalled downloads

## Synopsis

**TimeoutStalled** [ `TimeoutStalled seconds`]

`Default`
TimeoutStalled 3600

`Context`
server config

`Module`
mod_core

`Compatibility`
1.1.6 and later

## Description

The TimeoutStalled directive sets the maximum number of seconds a data connection between the proftpd server and an FTP client can exist but have no actual data transferred (i.e. "stalled"). If the seconds argument is set to 0, data transfers are allowed to stall indefinitely.

## See also

## Examples

# TimesGMT

## Name

TimesGMT –– Toggle time display between GMT and local

## Synopsis

**TimesGMT** [ `TimesGMT on|off`]

`Default`
(versions 1.2.0pre9 and beyond) on
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`

## Description

Compatibility: 1.2.0pre9 and later The TimesGMT option causes the server to report all ls and MDTM times in GMT and not local time.

## See also

## Examples

# TransferLog

## Name

TransferLog –– Specify the path to the transfer log

## Synopsis

**TransferLog** [ `TransferLog filename` | `NONE`]

`Default`
: TransferLog /var/log/xferlog

`Context`
: server config, <Anonymous>, <VirtualHost>, <Global>

`Module`
: mod_core

`Compatibility`
: 1.1.4 and later

## Description

The TransferLog directive configures the full path to the "wu−ftpd style" file transfer log. Separate log files can be created for each Anonymous and/or VirtualHost. Additionally, the special keyword NONE can be used, which disables wu−ftpd style transfer logging for the context in which the directive is used (only applicable to version 1.1.7 and later). See Also: ExtendedLog, LogFormat

## See also

## Examples

# Umask

## Name

Umask –– Set the default Umask

## Synopsis

**Umask** [ `Umask file octal–mask [directory octal–mask]`]

`Default`
None
`Context`
server config, <Anonymous>, <VirtualHost>, <Directory>, <Global>, .ftpaccess
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

Umask sets the mask applied to newly created file and directory permissions within a given context. By default, the Umask in the server configuration, <VirtualHost> or <Anonymous> block is used, unless overridden by a "per–directory" Umask setting. Any arguments supplied must be an octal number, in the format 0xxx. An optional second argument can specify a Umask to be used when creating directories. If a second argument isn't specified, directories are created using the default Umask in the first argument. For more information on umasks, consult your operating system documentation/man pages.

Proftpd will not create files that have the execution bit turned on, this is a security driven design decision. The permissions of the uploaded file can be changed by issuing a SITE CHMOD command can be used to change the mode of the uploaded file. Syntax of the command is: SITE CHMOD <mode> <file>.

## See also

## Examples

# UseFtpUsers

## Name

UseFtpUsers –– Block based on /etc/ftpusers

## Synopsis

**UseFtpUsers** [ `UseFtpUsers on|off`]

`Default`
: UseFtpUsers on

`Context`
: server config, <Anonymous>, <VirtualHost>, <Global>

`Module`
: mod_core

`Compatibility`
: 0.99.0 and later

## Description

Legacy FTP servers generally check a special authorization file (typically /etc/ftpusers) when a client attempts to authenticate. If the user's name is found in this file, FTP access is denied. For compatibility sake, proftpd defaults to checking this file during authentication. This behavior can be suppressed using the UseFtpUsers configuration directive.

## See also

## Examples

# UseGlobbing

## Name

UseGlobbing -- Toggles use of glob() functionality

## Synopsis

**UseGlobbing** [ `on` | `off`]

`Default`
UseGlobbing on

`Context`
server config, <VirtualHost>, <Global>, <Anonymous>

`Module`
mod_ls

`Compatibility`
1.2.5rc1 and later

## Description

The UseGlobbing directive controls use of glob() functionality, which is needed for supporting wildcard characters such as *.

## See also

# User

## Name

User –– Set the user the daemon will run as

## Synopsis

**User** [ `User userid`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The User directive configures which user the proftpd daemon will normally run as. By default, proftpd runs as root which is considered undesirable in all but the most trustful network configurations. The User directive used in conjunction with the Group directive instructs the daemon to switch to the specified user and group as quickly as possible after startup. On some unix variants, the daemon will occasionally switch back to root in order to accomplish a task which requires super–user access. Once the task is completed, root privileges are relinquished and the server continues to run as the specified user and group. When applied to a <VirtualServer> block, proftpd will run as the specified user/group on connections destined for the virtual server's address or port. If either User or Group is applied to an <Anonymous> block, proftpd will establish an anonymous login when a user attempts to login with the specified userid, as well as permanently switching to the corresponding uid/gid (matching the User/Group parameters found in the anonymous block) after login. Note: When an authorized unix user is authenticated and logs in, all former privileges are released, the daemon switches permanently to the logged in user's uid/gid, and is never again capable of switching back to root or any other user/group.

## See also

## Examples

# UserAlias

## Name

UserAlias –– Alias a username to a system user

## Synopsis

**UserAlias** [ `UserAlias login-user userid`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_auth
`Compatibility`
0.99.0 and later

## Description

ProFTPD requires a real username/uid when authenticating users as provided by PAM, AuthUserFile or another authentication mechanism. There are however times when additional aliases are required but it is undesirable to provide additional login accounts.

UserAlias provides a mechanism to do this, a typical and common example is within Anonymous configuration blocks. It is normal for the server to use 'ftp' as the primary authentication user, however it is common practice for users to login using "anonymous". This is achieved by adding the following to the config file.

## See also

## Examples

UserAlias anonymous ftp

# UserDirRoot

## Name

UserDirRoot –– Set the chroot directory to a subdirectory of the anonymous server

## Synopsis

**UserDirRoot** [ `UserDirRoot on|off`]

`Default`
off
`Context`
<Anonymous>
`Module`
mod_auth
`Compatibility`
1.2.0pre2 and later

## Description

When set to true, the chroot base directory becomes a subdirectory of the anonymous ftp directory, based on the username of the current user. For example, assuming user "foo" is aliased to "ftp", logging in as "foo" causes proftpd to run as real user ftp, but to chroot into ~ftp/foo instead of just ~ftp.

## See also

## Examples

# UseReverseDNS

## Name

UseReverseDNS –– Toggle rDNS lookups

## Synopsis

**UseReverseDNS** [ `UseReverseDNS on|off`]

`Default`
UseReverseDNS on

`Context`
server config

`Module`
mod_core

`Compatibility`
1.1.7 and later

## Description

Normally, incoming active mode data connections and outgoing passive mode data connections have a reverse DNS lookup performed on the remote host's IP address. In a chroot environment (such as <Anonymous> or DefaultRoot), the /etc/hosts file cannot be checked and the only possible resolution is via DNS. If for some reason, DNS is not available or improperly configured this can result in proftpd blocking ("stalling") until the libc resolver code times out. Disabling this directive prevents proftpd from attempting to reverse–lookup data connection IP addresses.

## See also

## Examples

# UserOwner

## Name

UserOwner –– Set the user ownership of new files / directories

## Synopsis

**UserOwner** [ `UserOwner username`]

`Default`
None

`Context`
<Anonymous>, <Directory>

`Module`
mod_core

`Compatibility`
1.2pre11 and later

## Description

The UserOwner directive configures which user all newly created directories and files will be owned by, within the context that UserOwner is applied to. The user ID of username cannot be 0 (root). Where it is used, the GroupOwner directive is not restricted to groups that the current user is a member of.

## See also

## Examples

# UserPassword

## Name

UserPassword –– Creates a hardcoded username/password pair

## Synopsis

**UserPassword** [ `UserPassword userid hashed−password`]

`Default`
None
`Context`
server config, <VirtualHost>, <Anonymous>, <Global>
`Module`
mod_core
`Compatibility`
0.99.0pl5 and later

## Description

The UserPassword directive creates a password for a particular user which overrides the user's normal password in /etc/passwd (or /etc/shadow). The override is only effective inside the context to which UserPassword is applied. The hashed−password argument is a cleartext string which has been passed through the standard unix crypt() function. Do NOT use a cleartext password. This can be useful when combined with UserAlias to provide multiple logins to an Anonymous FTP site. See Also: GroupPassword

## See also

## Examples

# UserRatio

## Name

UserRatio –– Ratio directive

## Synopsis

**UserRatio** [ `UserRatio foo1 foo2 foo3`]

`Default`
: None known

`Context`
: <Directory>, <Anonymous>, <Limit>,.ftpaccess

`Module`
: mod_ratio

`Compatibility`
: at least 1.2.0 and later

## Description

The UserRatio directive .... Example: UserRatio

## See also

## Examples

# VirtualHost

## Name

VirtualHost –– Define a virtual ftp server

## Synopsis

**VirtualHost** [ `<VirtualHost address>`]

`Default`
None
`Context`
server config
`Module`
mod_core
`Compatibility`
0.99.0 and later

## Description

The VirtualHost configuration block is used to create an independent set of configuration directives that apply to a particular hostname or IP address. It is often used in conjunction with system level IP aliasing or dummy network interfaces in order to establish one or more "virtual" servers which all run on the same physical machine. The block is terminated with a </VirtualHost> directive. By utilizing the Port directive inside a VirtualHost block, it is possible to create a virtual server which uses the same address as the master server, but listens on a separate tcp port (incompatible with ServerType inetd). When proftpd starts, virtual server connections are handled in one of two ways, depending on the ServerType setting: inetd The daemon examines the destination address and port of the incoming connection handed off from inetd. If the connection matches one of the configured virtual hosts, the connection is serviced based on the appropriate configuration. If no virtual host matches, and the main server does not match, the client is informed that no server is available to service their requests and disconnected. standalone After parsing the configuration file, the daemon begins listening for connections on all configured ports, spawning child processes as necessary to handle connections for either the main server or any virtual servers. Because of the method that the daemon uses to listen for connections when in standalone mode, it is possible to support an exceedingly large number of virtual servers, potentially exceeding the number of per–process file descriptors. This is due to the fact that a single file descriptor is used to listen to each configured port, regardless of the number of addresses being monitored. Note that it may be necessary to increase the tcpBackLog value on heavily loaded servers in order to avoid kernel rejected client connections ("Connection refused").

## See also

# Examples

# WtmpLog

## Name

WtmpLog –– Toggle logging to wtmp

## Synopsis

**WtmpLog** [ `WtmpLog on|off|NONE`]

`Default`
WtmpLog on

`Context`
server config, <VirtualHost>, <Anonymous>, <Global>

`Module`
mod_core

`Compatibility`
1.1.7 and later

## Description

The WtmpLog directive controls proftpd's logging of ftp connections to the host system's wtmp file (used by such commands as `last'). By default, all connections are logged via wtmp. Please report any corrections or additions via http://bugs.proftpd.net/

## See also

## Examples

# Chapter 2. List of modules

# mod_auth

## Name

mod_auth –– Authentication module

## Synopsis

**mod_auth**

## Description

FIXME FIXME FIXME

## See also

DefaultChdir DefaultRoot LoginPasswordPrompt RootLogin UserAlias UserDirRoot

# mod_code

## Name

mod_code –– FIX ME FIX ME

## Synopsis

**mod_code**

## Description

FIXME FIXME FIXME

## See also

DisplayConnect DisplayFirstChdir

# mod_core

## Name

mod_core –– Core module

## Synopsis

**mod_core**

## Description

This module provides all the core functionality ProFTPD needs to function, this module must be compiled in.

## See also

AccessDenyMsg AccessGrantMsg Allow AllowAll AllowFilter AllowForeignAddress AllowGroup AllowOverwrite AllowRetrieveRestart AllowStoreRestart AllowUser AnonRequirePassword Anonymous AnonymousGroup AuthAliasOnly AuthUsingAlias Bind CDPath Class Classes CommandBufferSize DefaultServer DefaultTransferMode DeferWelcome DeleteAbortedStores Deny DenyAll DenyFilter DenyGroup DenyUser Directory DisplayGoAway DisplayLogin DisplayQuit Global Group GroupOwner GroupPassword HiddenStor HideGroup HideNoAccess HideUser IdentLookups IgnoreHidden Include Limit MasqueradeAddress MaxClients MaxClientsPerHost MaxHostsPerUser MaxInstances MaxLoginAttempts MultilineRFC2228 Order PassivePorts PathAllowFilter PathDenyFilter PidFile Port RequireValidShell RLimitCPU RLimitMemory RLimitOpenFiles ScoreboardPath ServerAdmin ServerIdent ServerName ServerType ShowSymlinks SocketBindTight SyslogFacility SyslogLevel tcpBackLog tcpNoDelay tcpReceiveWindow tcpSendWindow TimeoutIdle TimeoutLogin TimeoutNoTransfer TimeoutStalled TimesGMT TransferLog Umask UseFtpUsers User UseReverseDNS UserOwner UserPassword VirtualHost WtmpLog

# mod_ldap

## Name

mod_ldap –– LDAP authentication support

## Synopsis

**mod_ldap**

## Description

mod_ldap provides LDAP authentication support for ProFTPD. It supports many features useful in "toaster" environments such as default UID/GID and autocreation/autogeneration of home directories.

## See also

LDAPAuthBinds LDAPDefaultAuthScheme LDAPDefaultGID LDAPDefaultUID LDAPDNInfo LDAPDoAuth LDAPDoGIDLookups LDAPDoUIDLookups LDAPForceDefaultGID LDAPForceDefaultUID LDAPHomedirOnDemand LDAPHomedirOnDemandPrefix LDAPHomedirOnDemandPrefixNoUsername LDAPHomedirOnDemandSuffix LDAPNegativeCache LDAPQueryTimeout LDAPSearchScope LDAPServer LDAPUseTLS

# mod_log

## Name

mod_log –– Logging support

## Synopsis

**mod_log**

## Description

Logging support, including enhanced formatting options.

## See also

AllowLogSymlinks ExtendedLog LogFormat SystemLog

# mod_ls

## Name

mod_ls –– file listing functionality

## Synopsis

**mod_ls**

## Description

FIXME FIXME FIXME

## See also

DirFakeGroup DirFakeMode DirFakeUser LsDefaultOptions ShowDotFiles UseGlobbing

# mod_pam

## Name

mod_pam –– Pluggable authentication modules support

## Synopsis

**mod_pam**

## Description

FIXME FIXME FIXME

## See also

AuthPAM AuthPAMConfig

# mod_quota

## Name

mod_quota –– Module to implement per–user quotas

## Synopsis

**mod_quota**

## Description

FIXME FIXME FIXME

## Notes

mod_quota forces a quota recalculation when the .quota file is missing. It is therefore possible to tie normal shell logins into deleting the file to force recalcualtions.

## See also

DefaultQuota QuotaBlockName QuotaBlockSize QuotaCalc QuotaExempt Quotas QuotaType

# mod_ratio

## Name

mod_ratio –– FIX ME FIX ME

## Synopsis

**mod_ratio**

## Description

FIXME FIXME FIXME

## See also

AnonRatio ByteRatioErrMsg CwdRatioMsg FileRatioErrMsg GroupRatio HostRatio LeechRatioMsg RatioFile Ratios RatioTempFile SaveRatios UserRatio

# mod_readme

## Name

mod_readme -- "README" file support

## Synopsis

**mod_readme**

## Description

FIXME FIXME FIXME

## See also

DisplayReadme

# mod_sample

## Name

mod_sample –– Example module

## Synopsis

**mod_sample**

## Description

This module only provides an example set of code as a template for a budding module programmer.

## See also

FooBarDirective

# mod_site

## Name

mod_site -- FIX ME FIX ME

## Synopsis

**mod_site**

## Description

FIXME FIXME FIXME

## See also

AllowChmod

# mod_sql

## Name

mod_sql –– SQL support module

## Synopsis

**mod_sql**

## Description

This module provides the necessary support for SQL based authentication, logging and other features as required. It replaces the SQL modules which were shipped with 1.2.0rc2 and earlier.

## See also

MySQLInfo PostgresInfo PostgresPort SQLAuthenticate SQLAuthoritative SQLAuthTypes SQLConnectInfo SQLDefaultGID SQLDefaultHomedir SQLDefaultUID SQLDoAuth SQLDoGroupAuth SQLEmptyPasswords SQLEncryptedPasswords SQLGidField SQLGroupGIDField SQLGroupInfo SQLGroupMembersField SQLGroupnameField SQLGroupTable SQLGroupWhereClause SQLHomedir SQLHomedirField SQLHomedirOnDemand SQLLog SQLLogDirs SQLLogHits SQLLogHosts SQLLoginCountField SQLLogStats SQLMinID SQLMinUserGID SQLMinUserUID SQLNamedQuery SQLPasswordField SQLProcessGrEnt SQLProcessPwEnt SQLRatios SQLRatioStats SQLScrambledPasswords SQLShellField SQLShowInfo SQLSSLHashedPasswords SQLUidField SQLUserInfo SQLUsernameField SQLUserTable SQLUserWhereClause SQLWhereClause

# mod_unixpw

## Name

mod_unixpw –– UNIX style authentication methods

## Synopsis

**mod_unixpw**

## Description

This module supports the password file (/etc/passwd) style of authentication methods.

## See also

AuthGroupFile AuthPAMAuthoritative AuthUserFile PersistentPasswd

# mod_wrap

## Name

mod_wrap -- Interface to libwrap

## Synopsis

**mod_wrap**

## Description

It enables the daemon to use the common tcpwrappers access control library while in standalone mode, and in a very configurable manner. It is not compiled by default.

If not installed on your system, the TCP wrappers library, required by this module, can be found here, on Wietse Venema's site. Once installed, it highly recommended that the hosts_access(3) and hosts_access(5) man pages be read and understood.

Many programs will automatically add entries in the common allow/deny files, and use of this module will allow a ProFTPD daemon running in standalone mode to adapt as these entries are added. The portsentry program does this, for example: when illegal access is attempted, it will add hosts to the /etc/hosts.deny file.

## See also

TCPAccessFiles TCPAccessSyslogLevels TCPGroupAccessFiles TCPServiceName TCPUserAccessFiles

# mod_xfer

## Name

mod_xfer –– FIX ME FIX ME

## Synopsis

**mod_xfer**

## Description

FIXME FIXME FIXME

## See also

RateReadBPS RateReadFreeBytes RateReadHardBPS RateWriteBPS RateWriteFreeBytes RateWriteHardBPS

# Chapter 3. List of configuration contexts

# server config

## Name

server config –– server config

## Synopsis

**server config**

## Description

FIXME FIXME FIXME

## See also

# Global

## Name

Global –– Global

## Synopsis

**Global**

## Description

FIXME FIXME FIXME

## See also

# VirtualHost

## Name

VirtualHost –– VirtualHost

## Synopsis

**VirtualHost**

## Description

FIXME FIXME FIXME

## See also

# Anonymous

## Name

Anonymous –– Anonymous

## Synopsis

**Anonymous**

## Description

FIXME FIXME FIXME

## See also

AccessDenyMsg AccessGrantMsg AllowAll AllowChmod AllowFilter AllowForeignAddress AllowOverwrite AllowRetrieveRestart AllowStoreRestart AnonRatio AnonRequirePassword AuthAliasOnly AuthUsingAlias ByteRatioErrMsg CDPath CwdRatioMsg DefaultChdir DefaultQuota DeleteAbortedStores DenyAll DenyFilter Directory DirFakeGroup DirFakeMode DirFakeUser DisplayFirstChdir DisplayGoAway DisplayLogin DisplayQuit DisplayReadme ExtendedLog FileRatioErrMsg FooBarDirective Group GroupOwner GroupPassword GroupRatio HiddenStor HideGroup HideNoAccess HideUser HostRatio Include LDAPHomedirOnDemandPrefixNoUsername LeechRatioMsg Limit LoginPasswordPrompt LsDefaultOptions MaxClients MaxClientsPerHost MaxHostsPerUser PathAllowFilter PathDenyFilter QuotaBlockName QuotaBlockSize QuotaCalc QuotaExempt Quotas QuotaType RateReadBPS RateReadFreeBytes RateReadHardBPS RateWriteBPS RateWriteFreeBytes RateWriteHardBPS RatioFile Ratios RatioTempFile RequireValidShell RootLogin SaveRatios ShowDotFiles ShowSymlinks SQLAuthenticate SQLAuthoritative SQLDefaultHomedir SQLGidField SQLGroupGIDField SQLGroupInfo SQLGroupWhereClause SQLHomedir SQLHomedirField SQLLog SQLLogDirs SQLLogHits SQLLogHosts SQLLoginCountField SQLLogStats SQLMinUserGID SQLMinUserUID SQLNamedQuery SQLPasswordField SQLProcessGrEnt SQLProcessPwEnt SQLShowInfo SQLUidField SQLUserInfo SQLUsernameField SQLUserTable SQLUserWhereClause SyslogLevel TCPAccessFiles TCPAccessSyslogLevels TimesGMT TransferLog Umask UseFtpUsers UseGlobbing User UserAlias UserDirRoot UserOwner UserPassword UserRatio WtmpLog

# Limit

## Name

Limit –– Limit

## Synopsis

**Limit**

## Description

FIXME FIXME FIXME

## See also

Allow AllowAll AllowGroup AllowUser AnonRatio ByteRatioErrMsg CwdRatioMsg Deny DenyAll DenyGroup DenyUser FileRatioErrMsg FooBarDirective GroupRatio HostRatio IgnoreHidden LDAPHomedirOnDemandPrefixNoUsername LeechRatioMsg Order RatioFile Ratios RatioTempFile SaveRatios SQLAuthenticate SQLAuthoritative SQLDefaultHomedir SQLGidField SQLGroupGIDField SQLGroupInfo SQLGroupWhereClause SQLHomedir SQLHomedirField SQLLog SQLLogDirs SQLLogHits SQLLogHosts SQLLoginCountField SQLLogStats SQLMinUserGID SQLMinUserUID SQLNamedQuery SQLPasswordField SQLProcessGrEnt SQLProcessPwEnt SQLShowInfo SQLUidField SQLUserInfo SQLUsernameField SQLUserTable SQLUserWhereClause UserRatio

# .ftpaccess

## Name

.ftpaccess –– .ftpaccess

## Synopsis

**.ftpaccess**

## Description

FIXME FIXME FIXME

## See also

AllowAll AllowChmod AllowOverwrite AllowRetrieveRestart AllowStoreRestart AnonRatio ByteRatioErrMsg CwdRatioMsg DeleteAbortedStores DenyAll FileRatioErrMsg GroupOwner GroupRatio HostRatio LDAPHomedirOnDemandPrefixNoUsername LeechRatioMsg Limit RatioFile Ratios RatioTempFile SaveRatios SQLAuthenticate SQLAuthoritative SQLDefaultHomedir SQLGidField SQLGroupGIDField SQLGroupInfo SQLGroupWhereClause SQLHomedir SQLHomedirField SQLLog SQLLogDirs SQLLogHits SQLLogHosts SQLLoginCountField SQLLogStats SQLMinUserGID SQLMinUserUID SQLNamedQuery SQLPasswordField SQLProcessGrEnt SQLProcessPwEnt SQLShowInfo SQLUidField SQLUserInfo SQLUsernameField SQLUserTable SQLUserWhereClause Umask UserRatio