Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'e2fsck.conf.5' command

$ man e2fsck.conf.5

e2fsck.conf(5) File Formats Manual e2fsck.conf(5)

NAME

e2fsck.conf - Configuration file for e2fsck

DESCRIPTION

e2fsck.conf is the configuration file for e2fsck(8). It controls the default behavior of e2fsck(8) while it is checking ext2, ext3, or ext4 file systems.

The e2fsck.conf file uses an INI-style format. Stanzas, or top-level sections, are delimited by square braces: [ ]. Within each section, each line defines a relation, which assigns tags to values, or to a subsection, which contains further relations or subsections. An exam? ple of the INI-style format used by this configuration file follows be? low:

```
[section1]
    tag1 = value_a
    tag1 = value_b
    tag2 = value_c
[section 2]
    tag3 = {
        subtag1 = subtag_value_a
        subtag1 = subtag_value_b
        subtag2 = subtag_value_c
    }
```

```
        tag1 = value_d
        tag2 = value_e
    }
```

Comments are delimited by a semicolon (';') or a hash ('#') character at the beginning of the comment, and are terminated by the end of line character.

Tags and values must be quoted using double quotes if they contain spaces. Within a quoted string, the standard backslash interpretations apply: "\n" (for the newline character), "\t" (for the tab character), "\b" (for the backspace character), and "\\" (for the backslash character).

The following stanzas are used in the e2fsck.conf file. They will be described in more detail in future sections of this document.

[options]

This stanza contains general configuration parameters for e2fsck's behavior.

[defaults]

Contains relations which define the default parameters used by e2fsck(8). In general, these defaults may be overridden by command-line options provided by the user.

[problems]

This stanza allows the administrator to reconfigure how e2fsck handles various file system inconsistencies.

[scratch_files]

This stanza controls when e2fsck will attempt to use scratch files to reduce the need for memory.

THE [options] STANZA

The following relations are defined in the [options] stanza.

allow_cancellation

If this relation is set to a boolean value of true, then if the user interrupts e2fsck using ^C, and the file system is not explicitly flagged as containing errors, e2fsck will exit with an exit status of 0 instead of 32. This setting defaults to false.

accept_time_fudge

Unfortunately, due to Windows' unfortunate design decision to configure the hardware clock to tick localtime, instead of the more proper and less error-prone UTC time, many users end up in the situation where the system clock is incorrectly set at the time when e2fsck is run.

Historically this was usually due to some distributions having buggy init scripts and/or installers that didn't correctly de? tect this case and take appropriate countermeasures. Unfortu? nately, this is occasionally true even today, usually due to a buggy or misconfigured virtualization manager or the installer not having access to a network time server during the installa? tion process. So by default, we allow the superblock times to be fudged by up to 24 hours. This can be disabled by setting accept_time_fudge to the boolean value of false. This setting defaults to true.

broken_system_clock

The e2fsck(8) program has some heuristics that assume that the system clock is correct. In addition, many system programs make similar assumptions. For example, the UUID library depends on time not going backwards in order for it to be able to make its guarantees about issuing universally unique ID's. Systems with broken system clocks, are well, broken. However, broken system clocks, particularly in embedded systems, do exist. E2fsck will attempt to use heuristics to determine if the time can not be trusted; and to skip time-based checks if this is true. If this boolean is set to true, then e2fsck will always assume that the system clock can not be trusted.

buggy_init_scripts

This boolean relation is an alias for accept_time_fudge for backwards compatibility; it used to be that the behavior defined by accept_time_fudge above defaulted to false, and buggy_init_scripts would enable superblock time field to be

wrong by up to 24 hours. When we changed the default, we also renamed this boolean relation to accept_time_fudge.

clear_test_fs_flag

This boolean relation controls whether or not e2fsck(8) will offer to clear the test_fs flag if the ext4 file system is available on the system. It defaults to true.

defer_check_on_battery

This boolean relation controls whether or not the interval between file system checks (either based on time or number of mounts) should be doubled if the system is running on battery. This setting defaults to true.

indexed_dir_slack_percentage

When e2fsck(8) repacks a indexed directory, reserve the specified percentage of empty space in each leaf nodes so that a few new entries can be added to the directory without splitting leaf nodes, so that the average fill ratio of directories can be maintained at a higher, more efficient level. This relation defaults to 20 percent.

inode_count_fullmap

If this boolean relation is true, trade off using memory for speed when checking a file system with a large number of hard-linked files. The amount of memory required is proportional to the number of inodes in the file system. For large file systems, this can be gigabytes of memory. (For example a 40TB file system with 2.8 billion inodes will consume an additional 5.7 GB memory if this optimization is enabled.) This setting defaults to false.

log_dir

If the log_filename or problem_log_filename relations contains a relative pathname, then the log file will be placed in the directory named by the log_dir relation.

log_dir_fallback

This relation contains an alternate directory that will be used

if the directory specified by log_dir is not available or is not writable.

log_dir_wait

If this boolean relation is true, them if the directories speci? fied by log_dir or log_dir_fallback are not available or are not yet writable, e2fsck will save the output in a memory buffer, and a child process will periodically test to see if the log di? rectory has become available after the boot sequence has mounted the requested file system for reading/writing. This implements the functionality provided by logsave(8) for e2fsck log files.

log_filename

This relation specifies the file name where a copy of e2fsck's output will be written. If certain problem reports are sup? pressed using the max_count_problems relation, (or on a per-problem basis using the max_count relation), the full set of problem reports will be written to the log file. The filename may contain various percent-expressions (%D, %T, %N, etc.) which will be expanded so that the file name for the log file can in? clude things like date, time, device name, and other run-time parameters. See the LOGGING section for more details.

max_count_problems

This relation specifies the maximum number of problem reports of a particular type will be printed to stdout before further prob? lem reports of that type are squelched. This can be useful if the console is slow (i.e., connected to a serial port) and so a large amount of output could end up delaying the boot process for a long time (potentially hours).

no_optimize_extents

If this boolean relation is true, do not offer to optimize the extent tree by reducing the tree's width or depth. This setting defaults to false.

problem_log_filename

This relation specifies the file name where a log of problem

codes found by e2fsck be written. The filename may contain var? ious percent-expressions (%D, %T, %N, etc.) which will be ex? panded so that the file name for the log file can include things like date, time, device name, and other run-time parameters. See the LOGGING section for more details.

readahead_mem_pct

Use this percentage of memory to try to read in metadata blocks ahead of the main e2fsck thread. This should reduce run times, depending on the speed of the underlying storage and the amount of free memory. There is no default, but see readahead_kb for more details.

readahead_kb

Use this amount of memory to read in metadata blocks ahead of the main checking thread. Setting this value to zero disables readahead entirely. By default, this is set the size of two block groups' inode tables (typically 4MiB on a regular ext4 file system); if this amount is more than 1/50th of total physi? cal memory, readahead is disabled.

report_features

If this boolean relation is true, e2fsck will print the file system features as part of its verbose reporting (i.e., if the -v option is specified)

report_time

If this boolean relation is true, e2fsck will run as if the op? tions -tt are always specified. This will cause e2fsck to print timing statistics on a pass by pass basis for full file system checks.

report_verbose

If this boolean relation is true, e2fsck will run as if the op? tion -v is always specified. This will cause e2fsck to print some additional information at the end of each full file system check.

The following relations are defined in the [defaults] stanza.

undo_dir

This relation specifies the directory where the undo file should be stored. It can be overridden via the E2FSPROGS_UNDO_DIR en? vironment variable. If the directory location is set to the value none, e2fsck will not create an undo file.

THE [problems] STANZA

Each tag in the [problems] stanza names a problem code specified with a leading "0x" followed by six hex digits. The value of the tag is a subsection where the relations in that subsection override the default treatment of that particular problem code.

Note that inappropriate settings in this stanza may cause e2fsck to be? have incorrectly, or even crash. Most system administrators should not be making changes to this section without referring to source code.

Within each problem code's subsection, the following tags may be used:

description

This relation allows the message which is printed when this file system inconsistency is detected to be overridden.

preen_ok

This boolean relation overrides the default behavior controlling whether this file system problem should be automatically fixed when e2fsck is running in preen mode.

max_count

This integer relation overrides the max_count_problems parameter (set in the options section) for this particular problem.

no_ok This boolean relation overrides the default behavior determining whether or not the file system will be marked as inconsistent if the user declines to fix the reported problem.

no_default

This boolean relation overrides whether the default answer for this problem (or question) should be "no".

preen_nomessage

This boolean relation overrides the default behavior controlling

whether or not the description for this file system problem should be suppressed when e2fsck is running in preen mode.

no_nomsg

This boolean relation overrides the default behavior controlling whether or not the description for this file system problem should be suppressed when a problem forced not to be fixed, ei? ther because e2fsck is run with the -n option or because the force_no flag has been set for the problem.

force_no

This boolean option, if set to true, forces a problem to never be fixed. That is, it will be as if the user problem responds 'no' to the question of 'should this problem be fixed?'. The force_no option even overrides the -y option given on the com? mand-line (just for the specific problem, of course).

not_a_fix

This boolean option, it set to true, marks the problem as one where if the user gives permission to make the requested change, it does not mean that the file system had a problem which has since been fixed. This is used for requests to optimize the file system's data structure, such as pruning an extent tree.

THE [scratch_files] STANZA

The following relations are defined in the [scratch_files] stanza.

directory

If the directory named by this relation exists and is writeable, then e2fsck will attempt to use this directory to store scratch files instead of using in-memory data structures.

numdirs_threshold

If this relation is set, then in-memory data structures will be used if the number of directories in the file system are fewer than amount specified.

dirinfo

This relation controls whether or not the scratch file directory is used instead of an in-memory data structure for directory in?

formation. It defaults to true.

icount This relation controls whether or not the scratch file directory is used instead of an in-memory data structure when tracking in? ode counts. It defaults to true.

LOGGING

E2fsck has the facility to save the information from an e2fsck run in a directory so that a system administrator can review its output at their leisure. This allows information captured during the automatic e2fsck preen run, as well as a manually started e2fsck run, to be saved for posterity. This facility is controlled by the log_filename, log_dir, log_dir_fallback, and log_dir_wait relations in the [options] stanza. The filename in log_filename may contain the following percent-expres? sions that will be expanded as follows.

%d The current day of the month

%D The current date; this is a equivalent of %Y%m%d

%h The hostname of the system.

%H The current hour in 24-hour format (00..23)

%m The current month as a two-digit number (01..12)

%M The current minute (00..59)

%N The name of the block device containing the file system, with any directory pathname stripped off.

%p The pid of the e2fsck process

%s The current time expressed as the number of seconds since 1970-01-01 00:00:00 UTC

%S The current second (00..59)

%T The current time; this is equivalent of %H%M%S

%u The name of the user running e2fsck.

%U This percent expression does not expand to anything, but it sig? nals that any following date or time expressions should be ex? pressed in UTC time instead of the local timezone.

%y The last two digits of the current year (00..99)

%Y The current year (i.e., 2012).

EXAMPLES

The following recipe will prevent e2fsck from aborting during the boot process when a file system contains orphaned files. (Of course, this is not always a good idea, since critical files that are needed for the security of the system could potentially end up in lost+found, and starting the system without first having a system administrator check things out may be dangerous.)

```
[problems]
    0x040002 = {
        preen_ok = true
        description = "@u @i %i.  "
    }
```

The following recipe will cause an e2fsck logfile to be written to the directory /var/log/e2fsck, with a filename that contains the device name, the hostname of the system, the date, and time: e.g., "e2fsck-sda3.server.INFO.20120314-112142". If the directory containing /var/log is located on the root file system which is initially mounted read-only, then the output will be saved in memory and written out once the root file system has been remounted read/write. To avoid too much detail from being written to the serial console (which could poten? tially slow down the boot sequence), only print no more than 16 in? stances of each type of file system corruption.

```
[options]
    max_count_problems = 16
    log_dir = /var/log/e2fsck
    log_filename = e2fsck-%N.%h.INFO.%D-%T
    log_dir_wait = true
```

FILES

/etc/e2fsck.conf

The configuration file for e2fsck(8).

SEE ALSO

e2fsck(8)