NAME

selog − selective logging configuration

DESCRIPTION

Selog is a library of routines for error reporting, activity logging, and debug tracing. It gives users flexible control over which messages are written and where they are written to.

Messages are classified using “selectors” that determine their priority level (e.g. debug or info or error, etc.) and category (the part of the program that they relate to). You can selectively enable or disable messages based on their selectors using a simple textual configuration. In some cases you may be given fine control over the verbosity of a message, beyond just switching it on or off, using separate selectors that control optional parts of the message.

Enabled messages are written to one or more channels, as specified by the configuration. Several kinds of channel are supported, including the standard error stream, syslog, files, etc. You can configure selog to write messages with different selectors to different sets of output channels.

This manual describes how you can configure selog, and the behaviour of selog’s output channels. The programmer’s interface to selog is described in **selog**(3), as are the auxiliary libraries that can be used to replace the standard **err**(3) and **syslog**(3) interfaces with implementations based on selog.

CONFIGURATION

Given a selector defined by the program, selog needs to determine (for message selectors) where messages controlled by the selector are to be written, or (for option selectors) whether the selector is enabled or not. To do this, it scans the configuration string provided by the user, which comprises a sequence of selection items and channel items.

The selector has a notional on/off state which is manipulated as the configuration string is scanned. The initial state of a message selector is on iff its level is “info” or above. (That is, there is an implicit “+default” at the start of the configuration string.) Option selectors can start in either state.

If the selector matches a selection item, it is switched on or off according to whether the first character of the item is “+” or “−”. If the selector does not match a selection item then its state remains unchanged.

If the selector is on when a channel item is encountered, then messages controlled by the selector are written to the channel specified by the item. If there are trailing selection items not followed by a channel item, then selog appends an implicit “@stderr” channel item.

An option selector is enabled if it is enabled for any channel, and disabled only if it is disabled for all channels. This implies that it is best to group option selection items at the start of the configuration string, so that turning off a default-on option works as expected.

Syntax

The formal syntax is described using ABNF (RFC 4234).

Items may be terminated by semicolons, which are optional except between a channel item and a following selection item. White space may be freely inserted between tokens. Space is only required to separate the arguments of a channel item.

config = *SP* *(**selection* **channel* ";") **selection* **channel*

space = %x20 / %x09 / %x0A / %x0D
SP = **space*
SEP = 1**space*
TERM = *(*space* / ";")

Selection items

A selection item starts with a “+” if it is to enable selectors, or a “−” if it is to disable them. If this is followed by a category name then only selectors with the given name will match.

The category name may be followed by a level comparison. If the operator is “<” then the selector matches if its level is less than or equal to the given level. If the operator is “=” then the selector matches if its level is the same as the given level. If the operator is “>” or “.” then the selector matches if its level is greater than or equal to the given level. If the comparison is omitted then it is equivalent to “>all”.

If the category name is omitted, any selector matches if its level matches. In this case a “>”

comparison operator can be omitted.

```
selection   =   sign category compare level TERM
            /   sign category TERM
            /   sign compare level TERM
            /   sign level TERM

sign        =   ("+" / "−") SP
compare     =   ("<" / "=" / ">" / ".") SP

            ;   In order of increasing priority:
level       =   "trace"
            /   "debug"
            /   "all" / "option" / "option_off" / "opt_off"
            /   "verbose"
            /   "default" / "option_on" / "opt_on"
            /   "info"
            /   "notice"
            /   "warning" / "warn"
            /   "error" / "err"
            /   "critical" / "crit"
            /   "alert"
            /   "emergency" / "emerg"
            /   "fatal" / "exit"
            /   "abort"

category    =   *catchar SP
catchar     =   %x21-2A / %x2C / %x2F-3A / %x3F / %x41-7E / %x80-FF
            ;   all visible characters except + − . ; < = > @
```

Option selectors are distinguished from message selectors by having one of two special levels. The "default" level is used for options that are on by default, and the "option" level is used for options that are off by default. So, for example, you can turn on all options without turning on "verbose" messages using the selection item "+=option". The level name "all" is a synonym for "option" that is useful when you want to enable all non-debug message selectors and all option selectors. Message selectors with a higher level than "default" are also on by default.

Channel items

A channel item starts with an "@" and a keyword indicating what kind of channel it is. If the keyword is not recognised then if it starts with a "/" it is treated as the first argument following a "file" keyword, and if it starts with a "|" the rest is treated as the first argument following a "pipe" keyword. (In other words, "file" keywords can be omitted, and "|" is a synonym for "pipe".) The keyword may be followed by space-separated arguments as required. The details of each kind of channel are described in the next section.

```
channel     =   "@" SP kind *(SEP argument) TERM

kind        =   "file"
            /   "pipe"
            /   "rotate"
            /   "stderr"
            /   "stdout"
            /   "syslog"
            /   "|" argument
            /   argument

argument    =   *argchar
argchar     =   %x21-3A / %x3C-3F / %x41-7E / %x80-FF
            ;   all visible characters except ; @
```

CHANNELS

The following sub-sections describe the details of each kind of channel, the arguments they require after their keywords in the configuration string, and any differences from the usual message format described in the next section.

stderr

The *stderr* channel writes its messages to the standard error stream (file descriptor 2). It takes no arguments. The time stamp and host name are omitted.

stdout

The *stdout* channel writes its messages to the standard output stream (file descriptor 1) and is otherwise the same as the *stderr* channel.

file

The *file* channel's first argument is an absolute path to the log file. (Because arguments are separated with white space, the path name cannot contain spaces.) It takes an optional second argument which is the numeric access mode to be used when creating the file. The default mode is 0666. The mode is affected by the process's umask setting.

Messages are appended in a way that is compatible with multiple programs logging to the same file concurrently. Selog automatically re-opens the file when it detects that it has been moved aside, so there is no need to HUP the process(es) when the log is rotated. If the **log_rotate** option is enabled then the *file* channel behaves like the *rotate* channel.

The host name is ommitted from the messages.

rotate

The *rotate* channel is very similar to the file channel, except that it automatically opens a new log file at midnight. The given file name has a date stamp appended in the format "**.YYYY-MM-DD**". By default midnight is local time, unless the **log_zulu** option is enabled.

pipe

The arguments to the *pipe* channel are a command-line that is passed to the shell to start a sub-process. Messages are written to the sub-process's standard input stream via a pipe.

syslog

The *syslog* channel takes a single argument which is a syslog facility name, i.e. one of:

user	**uucp**	**local1**	**local0**
mail	**cron**	**local2**	
daemon	**authpriv**	**local3**	
auth	**ftp**	**local4**	
syslog	**ntp**	**local5**	
lpr	**security**	**local6**	
news	**console**	**local7**	

Messages are written to the system log socket using the facility specified by the channel configuration and a severity derived from the message selector. Most selog levels map directly to syslog severities. Those that do not map as follows:

SELOG_TRACE	LOG_DEBUG
SELOG_VERBOSE	LOG_INFO
SELOG_FATAL	LOG_CRIT
SELOG_ABORT	LOG_ALERT

The time stamp is in the standard syslog format and is in local time unless the **log_zulu** option is set. The host name is omitted because **syslogd**(8) adds it when necessary.

MESSAGE FORMAT

As well as determining where messages are written, the channel type also affects the format of the message prefix. For example, syslog has its own time stamp format. Messages written by selog follow the same general scheme, though parts are often omitted depending on the particular message and the channel it is written to.

time host prog[pid] file:line func() name level: message: error

time The time the message was written. Except for syslog, the format is similar to ISO 8601 / RFC 3339. (For readability the "T" separator is replaced with a space and a space is added before

the time zone).
YYYY-MM-DD hh:mm:ss.fff +ZZ:ZZ

YYYY	year	2008
MM	month	04
DD	day	30
hh	hour	23
mm	minute	01
ss	second	59
fff	fraction	.500
ZZZZ	time zone	+01:00

Fractional seconds are usually omitted. They can be enabled to millisecond precision using the **log_msec** option, and to microsecond precision using the **log_usec** option.

By default the time stamp is in local time. Positive time zone offsets are ahead of UTC (East of Greenwich). You can choose UTC time stamps using the **log_zulu** option, in which case the time zone is written as "Z" instead of as a numeric offset. You can omit the time zone by turning off the **log_tz** option, though this is unwise especially if you are logging local time in a place which is subject to daylight saving variations.

host The fully-qualified host name of the machine that generated the message.

prog[pid]
The program name and process ID. The pid field is turned on using the **log_pid** option.

The precise format of the preceding fields, and whether or not they are present, depends on the channel, as described in the previous section. The presence of the following fields depends on the message.

file:line
The source code file and line number where the message is generated. This is usually omitted except by "trace" messages.

func() The source function where the message is generated. This is usually omitted except by "trace" messages.

func() The source function where the message is generated. This is usually omitted except by "trace" messages.

name The message selector's category name.

level The message selector's level.

mesage The main part of the message, which has no particular format.

error Operating system errors are generally added to the end of the message after a colon.

DIAGNOSTICS

This section lists the built-in selectors used by selog itself. Selectors are written **{***category***,** *LEVEL***}.**

{log_config, ERROR}
This is used by selog to report configuration syntax errors. Because it is used before selog is fully initialized, the messages it controls are always written to the standard error stream.

{log_msec, OPTION_OFF}
For logging time stamps to millisecond accuracy. Overridden by *log_usec*.

{log_panic, FATAL}
If an error occurs when selog is writing a message, it tries to write the error using the *log_panic* selector before exiting the program.

{log_pid, OPTION_OFF}
For enabling the process ID field.

{log_rotate, OPTION_OFF}
For enabling auto-rotation of *file* channels.

{log_tz, OPTION_ON}
For disabling the time zone field. This is unwise especially if you are logging local time in a place which is subject to daylight saving variations.

{log_usec, OPTION_OFF}
For logging time stamps to millisecond accuracy. Overrides *log_msec*.

{log_zulu, OPTION_OFF}
For logging in UTC instead of local time.

ENVIRONMENT

SELOG_CONFIG
Overrides the configuration string passed to selog by the program.

SEE ALSO

selog(3), **syslogd**(8)

RFC 5234: Augmented BNF for Syntax Specifications: ABNF,
by Dave Crocker (ed.), Paul Overell, January 2008.

RFC 3339: Date and time on the Internet: Timestamps,
by Graham Klyne (ed.), Chris Newman, July 2002.

AUTHOR

Written by Tony Finch <dot@dotat.at> <fanf2@cam.ac.uk>
at the University of Cambridge Computing Service.
Source available from <http://dotat.at/prog/selog>