

SysLog

Petri Nordlund

Copyright © CopyrightÂ©1995 Petri Nordlund. All rights reserved.

COLLABORATORS

	<i>TITLE :</i> SysLog		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Petri Nordlund	July 20, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	SysLog	1
1.1	Main menu	1
1.2	Copyright	1
1.3	Acknowledgments	3
1.4	Introduction	3
1.5	Facilities and levels	3
1.6	Developers	4
1.7	Author	7
1.8	PGP KEY	8
1.9	History	8
1.10	SysLogDaemon	8
1.11	SysLogDaemon options	10
1.12	LOG:	11
1.13	logger	11
1.14	logger options	12
1.15	newsyslog	13
1.16	newsyslog options	15

Chapter 1

SysLog

1.1 Main menu

S y s L o g

Copyright

Acknowledgments

Introduction

Developers

Author

History

SysLogDaemon

logger

LOG:

newsyslog

by Petri Nordlund

1.2 Copyright

COPYRIGHT

~~~~~

Copyright © 1995 Petri Nordlund.

Copyright (c) 1983, 1988 Regents of the University of California.  
All rights reserved.

Redistribution and use in source and binary forms, with or without  
modification, are permitted provided that the following conditions  
are met:

1. Redistributions of source code must retain the above copyright

notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright 1988, 1989 by the Massachusetts Institute of Technology

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of M.I.T. and the M.I.T. S.I.P.B. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. and the M.I.T. S.I.P.B. make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

#### DISCLAIMER

~~~~~

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

TRADEMARKS

~~~~~

Amiga and Workbench are trademarks of Amiga Technologies GmbH.

UNIX is a trademark of X/Open Company Ltd.

## 1.3 Acknowledgments

### ACKNOWLEDGMENTS

~~~~~

SysLog is based on NetBSD syslog software developed by the University of California, Berkeley and its contributors. Newsyslog was developed by Theodore Ts'o, of the Massachusetts Institute of Technology.

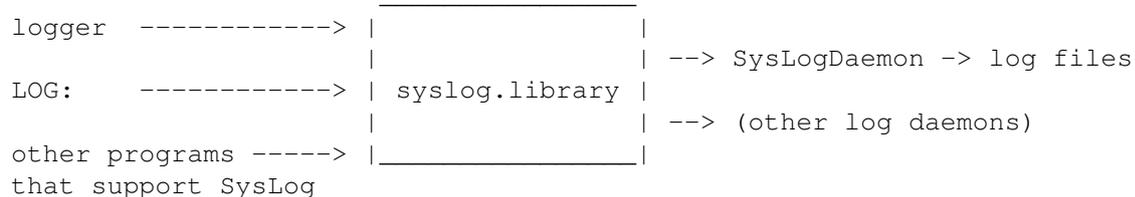
Code has been modified and improved to make it work on Amiga.

1.4 Introduction

INTRODUCTION

~~~~~

SysLog is a system that let's you log messages from various sources to various destinations. Here's a small diagram of SysLog:



Logger is a program that let's users to log messages to the system log. LOG: is a device that redirects everything written to it to the system log.

If the SysLogDaemon (or another log daemon) isn't running, all log messages are discarded. SysLogDaemon is usually started when the system is booted.

The newsyslog program is used to maintain log files. It can be configured to automatically archive old log files.

## 1.5 Facilities and levels

### FACILITIES AND LEVELS

~~~~~

Each message has a facility and a level. Facility tells us who

generated the message, and level indicates how important the message is.

Facility and level always form a pair:

```
facility.level
```

The facility is separated from the level by a period. For example:

```
user.notice
kern.debug
mail.info
```

FACILITIES

kern	kernel/system messages
user	user-level messages
mail	mail system
daemon	system daemons/commodities
auth	security/authorization messages
syslog	messages generated internally by SysLogDaemon
news	network news subsystem
uucp	UUCP sybssystem
cron	cron utility
authpriv	security/authorization messages (private)
ftp	ftp daemon
mark	internal mark messages generated by SysLogDaemon
local0	reserved for local use
local1	reserved for local use
local2	reserved for local use
local3	reserved for local use
local4	reserved for local use
local5	reserved for local use
local6	reserved for local use
local7	reserved for local use

LEVELS

emerg	indicates emergency situation, system is unusable
alert	action must be taken immediately
crit	critical condition
err	error condition
warning	warning condition
notice	normal but significant condition
info	informational
debug	debug-level messages

1.6 Developers

INFORMATION TO DEVELOPERS

~~~~~

It's very easy to support SysLog. All you need to do is to link your

application with the C link library (SAS/C or GCC). Then you can just call the SysLog()-function to log messages. You can also use the syslog.library directly.

You have to copy the include files in Developer drawer to correct directories. If you're using SAS/C, then just copy the contents of the include-directory to SC:include. If you're using GCC, copy to GCC:os-include. You'll also have to copy the contents of GCC:os-include directory to GCC:os-include.

There are a few example programs that will demonstrate how to use SysLog.

#### SYSLOG.LIBRARY

See the Developer directory for SysLog.autodoc.

Programs that want to listen to log messages are called spies. The syslog.library redirects all messages to all spies. If there are no spies, the messages are discarded.

#### THE C LINK LIBRARY

These functions are provided in the C link library:

```
void SysLog(LONG priority, STRPTR message, ...)
void VSysLog(LONG priority, STRPTR message, va list args)
void OpenLog(STRPTR tag, LONG options, LONG facility)
void CloseLog(void)
int SetLogMask(LONG mask)
```

The SysLog() function writes message to syslog.library, which redirects it to all spies.

The message is identical to a printf format string. Note that the '%m' is not supported, although it is in the UNIX version. A trailing newline is added if none is present. Maximum message length including time is 1024 characters.

The VSysLog() function is an alternate form in which the arguments have already been captured using the variable-length argument facilities of varargs.

The message is tagged with priority. Priorities are encoded as a facility and a level. The facility describes the part of the system generating the message. The level is selected from the following ordered (high to low) list:

|           |                                                                                        |
|-----------|----------------------------------------------------------------------------------------|
| LOG_EMERG | A panic condition.                                                                     |
| LOG_ALERT | A condition that should be corrected immediately, such as a corrupted system database. |

|             |                                                                                     |
|-------------|-------------------------------------------------------------------------------------|
| LOG_CRIT    | Critical conditions, e.g., hard device errors.                                      |
| LOG_ERR     | Errors.                                                                             |
| LOG_WARNING | Warning messages.                                                                   |
| LOG_NOTICE  | Conditions that are not error conditions, but should possibly be handled specially. |
| LOG_INFO    | Informational messages.                                                             |
| LOG_DEBUG   | Messages that contain information normally of use only when debugging a program.    |

The `OpenLog()` function provides for more specialized processing of the messages sent by `SysLog()` and `VSysLog()`. The parameter `ident` is a string that will be prepended to every message. The `options` argument is a bit field specifying logging options, which is formed by OR'ing one or more of the following values:

|            |                                                                                                                                                 |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| LOG_CONS   | If <code>SysLog()</code> cannot pass the message to any spy it will attempt to write the message to the console.                                |
| LOG_PERROR | Write the message to standard error output as well to the system log.                                                                           |
| LOG_PID    | Log the process id with each message. The process number is used here if Executive isn't running, in which case PID is obtained from Executive. |

IMPORTANT! The `LOG_CONS` and `LOG_PERROR` options can't be used when `SysLog()` or `VSysLog()` is called from a task. `LOG_PID` can't be used when calling from an interrupt or when multitasking is disabled.

The `facility` parameter encodes a default facility to be assigned to all messages that do not have an explicit facility encoded:

|              |                                                                                   |
|--------------|-----------------------------------------------------------------------------------|
| LOG_AUTH     | The authorization system: login etc.                                              |
| LOG_AUTHPRIV | The same as LOG AUTH, but logged to a file readable only by selected individuals. |
| LOG_CRON     | Cron utility                                                                      |
| LOG_DAEMON   | System daemons/commodities.                                                       |
| LOG_FTP      | Ftp daemon.                                                                       |
| LOG_KERN     | Messages generated by the system.                                                 |
| LOG_MAIL     | The mail system.                                                                  |
| LOG_NEWS     | The network news system.                                                          |
| LOG_SYSLOG   | Messages generated internally by SysLogDaemon.                                    |
| LOG_USER     | Messages generated by user processes. This is the                                 |

---

default facility identifier if none is specified.

|            |                                                                      |
|------------|----------------------------------------------------------------------|
| LOG_UUCP   | The uucp system.                                                     |
| LOG_LOCAL0 | Reserved for local use. Similarly for LOG_LOCAL1 through LOG_LOCAL7. |

The CloseLog() function is used after calling OpenLog().

The SetLogMask() function sets the log priority mask and returns the previous mask. Calls to SysLog() with a priority not set in the new mask are rejected. The mask for an individual priority pri is calculated by the macro LOG\_MASK(pri); the mask for all priorities up to and including toppri is given by the macro LOG\_UPTO(toppri);. The default allows all priorities to be logged. Of course SetLogMask() only applies to the current program.

#### EXAMPLES

```
SysLog(LOG_ALERT, "who: internal error 23");

OpenLog("ftpd", LOG_PID, LOG_FTP);
SetLogMask(LOG_UPTO(LOG_ERR));
SysLog(LOG_INFO, "Connection from host %d", CallingHost);
```

## 1.7 Author

#### AUTHOR

#### Email

```
petrin@megabaud.fi
petrin@sik.ppoy.fi
```

#### WWW

SysLog WWW page:

<http://www.megabaud.fi/~petrin/SysLog.html>

My homepage:

<http://www.megabaud.fi/~petrin>

#### Snail mail

```
Petri Nordlund
Vanhamaantie 4
28800 PORI
FINLAND
```

#### Telephone

+358-39-6480 322 (EET)

---

PGP

For secure communication, please use my PGP key.

## 1.8 PGP KEY

PGP key

~~~~~

Below you will find my public key. Use a text editor to save it to disk and follow the instructions on PGP manual for using it to send encrypted email.

You can also obtain my PGP key by fingering one of my email addresses.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6ui
```

```
mQCNAjBz1LwAAAAEALP5+1K/Mqbr7VDFrsL/W3Srp1hRfp5GbLid6Gn38/Zfb1PJ
hb/yDAuJX0G9nZ/hMBs7nXPVuUPZsf1iEeOuDW6U3MiqOrhau7mi6P05KqTW/xYY
fzTeZ+K6relfoR0+ScV4i0fYpMi/O4n+3uZGVGc4X/JSVQyHsTDgjUTe/EdpAAUR
tCNQZXRYaSBOb3JkbHVuZCA8cGV0cmluQG1lZ2FiYXVkJmZpPg==
=EBE0
-----END PGP PUBLIC KEY BLOCK-----
```

1.9 History

HISTORY

~~~~~

V1.00

~~~~~

First public release.

1.10 SysLogDaemon

SYSLOGDAEMON

~~~~~

Write log messages to disk/screen/serial port.

OPTIONS

DESCRIPTION

SysLogDaemon writes log messages to destinations specified in a configuration file. A destination can be:

---

- file
- console
- serial port

The configuration file, SysLogDaemon.conf is search first from directory where the program executable is and then from S: directory.

The configuration file consists of lines with two fields: the selector field which specifies the types of messages and priorities to which the line applies and an action field which specifies the action to be taken if a message matches the selection criteria. The selector field is separated from the action field by one or more tab characters.

The selectors function are encoded as a facility, a period ("."), and a level, with no intervening white-space. Both the facility and the level are case insensitive.

The facility describes the part of the system generating the message. The level describes the severity of the message.

For further descriptions of both the facility and level keywords and their significance, see a list of facilities and levels.

If a message matches the specified facility and is of the specified level (or a higher level), the action specified in the action field will be taken.

Multiple selectors may be specified for a single action by separating them with semicolon (";") characters. It is important to note, however, that each selector can modify the ones preceding it.

Multiple facilities may be specified for a single level by separating with comma (",") characters.

An asterisk ("\*") can be used to specify all facilities or all levels.

The special facility{UI} "mark" receives a message at level "info" every 20 minutes. This is not enabled by a facility field containing an asterisk.

The special level "none" disables a particular facility.

The action field of each line specifies the action to be taken when the selector field selects a message. There are three actions:

- FILE
- CON
- SERIAL

Examples:

```
FILE=T:debug.log
FILE=s:messages
```

```
CON                               (use default CON: 0/0/640/100/Log)
CON=0/0/640/80/SysLogDaemon output
```

```
SERIAL                (serial.device, unit 0)
SERIAL=artser.device  (artser.device, unit 0)
SERIAL=artser.device/2 (artser.device, unit 2)
```

Files are not kept open by SysLogDaemon, so you can read them and edit them while SysLogDaemon is running. If you want a CON-window that can be closed, use this action:

```
FILE=CON:20/40/600/140/===CRITICAL SITUATION===/AUTO/WAIT
```

Only one message will be printed to this window. You can close the window when you have read the message.

Serial transfer is done at 9600bps 8 bits, no parity, one stop bit.

Blank lines and lines whose first non-blank character is a hash ("##") character are ignored.

A configuration file might appear as follows:

```
# Log all messages from cron to one file
cron.*                                FILE=SYS:cron.log

# Log all kernel/system messages, authentication messages of
# level notice or higher and anything of level err or higher
# to a console.
# Don't log private authentication messages!
*.err;kern.*;auth.notice;authpriv.none      CON

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none             FILE=s:messages

# Log all authpriv-level messages to a secure drive
authpriv.*                                FILE=SECURE:authpriv.log
```

The effects of multiple selectors are sometimes not intuitive. For example "mail.crit,\*.err" will select "mail" facility messages at the level of "err" or higher, not at the level of "crit" or higher.

## 1.11 SysLogDaemon options

```
SYSLOGDAEMON OPTIONS
```

```
~~~~~
CONFIG
```

```
Template: -f=CONFIG/K
Tooltype: CONFIG
Default: PROGDIR:SysLogDaemon.conf, S:SysLogDaemon.conf
```

Name of the configuration file.

---

## MARK

Template: -m=MARK/N/K  
Tooltype: MARK  
Default: PROGDIR:SysLogDaemon.conf, S:SysLogDaemon.conf

Number of minutes between "mark" messages. The default is 20 minutes.

## QUIT

Template: -q=QUIT/S  
Tooltype: QUIT

Quit SysLogDaemon if it's running.

## VERSION

Template: -v=VERSION/S  
Tooltype: -

Display version number.

## 1.12 LOG:

## LOG:

~~~~~

LOG: device.

## DESCRIPTION

Everything that you write to LOG: device is logged to system log.  
LOG: can be very useful when you want to log the output of some program, for example a cron utility.

For example:

Cron >LOG:

You can also specify a priority and tag for all messages:

Cron >LOG:cron.info/cron  
Cron >LOG:cron.info  
Cron >LOG:/cron

Here cron.info is the priority and cron is the tag.

## 1.13 logger

## LOGGER

~~~~~

---

Make entries in the system log.

#### OPTIONS

#### DESCRIPTION

Logger let's users log messages to system log. For example:

```
logger System rebooted
```

Creates a message like this:

```
Nov 03 18:42:24 logger: System rebooted
```

A priority and a tag can be given:

```
logger PRI=user.notice TAG=User Test message
```

Produces:

```
Nov 03 18:43:05 User: Test message
```

If no message is given, message will be read from standard input.  
CTRL-D terminates input.

#### EXAMPLES

```
logger Test message
```

```
logger PRI=user.notice TAG=User Test message
```

## 1.14 logger options

#### LOGGER OPTIONS

~~~~~

##### FILE

```
Template: -f=FILE/K
Tooltype: FILE
```

Log the specified file.

##### PRI

```
Template: -p=PRI/K
Tooltype: PRI
```

Enter the message with specified priority. The priority may be specified numerically or as a "facility.level" pair.

See the list of facilities and levels.

---

## TAG

Template: -t=TAG/K  
Tooltype: TAG

Mark every message with the specified tag.

## PID

Template: -p=PID/K  
Tooltype: PID

Log the process number of the logger process with each line.

SysLog obtains the PID from Executive, if it's running.

## STDERR

Template: -s=STDERR/K  
Tooltype: STDERR

Log the message to standard error, as well as the system log.

## VERSION

Template: -v=VERSION/S  
Tooltype: -

Display version number.

## MESSAGE

Template: MESSAGE/F  
Tooltype: MESSAGE

Write this message to log. If not specified, and the FILE-option is not provided, standard input is logged.

## 1.15 newsyslog

## NEWSYSLOG

~~~~~

Maintain system log files to manageable sizes.

## OPTIONS

## DESCRIPTION

Newsyslog is a program that should be scheduled to run periodically by a cron utility. When it is executed it archives log files if necessary. If a log file is determined to require archiving, newsyslog rearranges the files so that "logfile" is empty, "logfile.0" has the last period's

---

logs in it, "logfile.1" has the next to last period's logs in it, and so on, up to a user-specified number of archived logs. Optionally the archived logs can be compressed to save space.

A log can be archived because of two reasons. The log file can have grown bigger than a preset size in kilo-bytes, or a preset number of hours may have elapsed since the last log archive. The granularity of newsyslog is dependent on how often it is scheduled to run by cron. Since the program is quite fast, it may be scheduled to run every hour without any ill effects.

When starting up, newsyslog reads in a configuration file to determine which logs should be looked at. By default, this configuration file is searched from the directory where the program executable is. The file is also searched from S: directory. The configuration file is called "newsyslog.conf". Each line of the file contains information about a particular log file that should be handled by newsyslog. Each line has four mandatory fields and one optional field, with a whitespace separating each field. Blank lines or lines beginning with "#" are ignored. The fields of the configuration file are as follows:

```
logfile name
number of archives
size of archives
archive interval
flags (optional)
```

The logfile name entry is the name of the system log file to be archived.

The number of archives entry specifies the number of archives to be kept besides the log file itself.

When the size of the logfile reaches size of archives, the logfile becomes trimmed as described above. If this field is replaced by a "\*", then the size of the logfile is not taken into account when determining when to trim the log file.

The number of hours entry specifies the time separation between the trimming of the log file. If this field is replaced by a "\*", then the number of hours since the last time the log was trimmed will not be taken into consideration.

The flags field specifies if the archives should have any special processing done to the archived log files. The "Z" flag will make the archive files compressed to save space using gzip, which has to be in the system search path. The "B" flag means that the file is a binary file, and so the ASCII message which newsyslog inserts to indicate the fact that the logs have been turned over should not be included.

#### EXAMPLES

Just show what would be done:

---

newsyslog NOACTION VERBOSE

## 1.16 newsyslog options

### NEWSYSLOG OPTIONS

~~~~~

#### CONFIG

Template: -f=CONFIG/K  
Tooltype: CONFIG  
Default:  PROGDIR:newsyslog.conf, S:newsyslog.conf

Name of the configuration file.

#### NOACTION

Template: -n=NOACTION/S  
Tooltype: NOACTION

Causes newsyslog not to trim the logs, but to print out what it would do if this option were not specified.

#### VERBOSE

Template: -v=VERBOSE/S  
Tooltype: VERBOSE

Places newsyslog in verbose mode. In this mode it will print out each log and its reasons for either trimming that log or skipping it.

#### VERSION

Template: -ver=VERSION/S  
Tooltype: -

Display version number.

---