

```

*****
7903 Mon Nov 11 13:47:11 2013
new/usr/src/cmd/logger/logger.c
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22  * Copyright (c) 2013 Gary Mills
23  * Copyright 2007 Sun Microsystems, Inc. All rights reserved.
24  * Use is subject to license terms.
25  */

27 /*      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T */
28 /*      All Rights Reserved      */

30 /*
31  * University Copyright- Copyright (c) 1982, 1986, 1988
32  * The Regents of the University of California
33  * All Rights Reserved
34  *
35  * University Acknowledgment- Portions of this document are derived from
36  * software developed by the University of California, Berkeley, and its
37  * contributors.
38  */

39 #pragma ident      "%Z%M% %I%      %E% SMI"

41 #include <sys/types.h>
42 #include <unistd.h>
43 #include <stdio.h>
44 #include <syslog.h>
45 #include <ctype.h>
46 #include <stdlib.h>
47 #include <string.h>
48 #include <locale.h>
49 #include <limits.h>
50 #include <pwd.h>
51 #include <errno.h>

53 #define LOG_MARK      (LOG_NFACILITIES << 3) /* mark "facility" */
54 #define LOGGER_BUFLEN 1024

56 struct code {
57     char      *c_name;
58     int      c_val;
59 };

```

```

76 static struct code      FacNames[] = {
77     "kern",      LOG_KERN,
78     "user",      LOG_USER,
79     "mail",      LOG_MAIL,
80     "daemon",    LOG_DAEMON,
81     "auth",      LOG_AUTH,
82     "security",  LOG_AUTH,
83     "mark",      LOG_MARK,
84     "syslog",    LOG_SYSLOG,
85     "lpr",       LOG_LPR,
86     "news",      LOG_NEWS,
87     "uucp",      LOG_UUCP,
88     "bsdcron",   LOG_BSDCRON,
89     "authpriv",  LOG_AUTHPRIV,
90     "ftp",       LOG_FTP,
91     "ntp",       LOG_NTP,
92     "audit",     LOG_AUDIT,
93     "console",   LOG_CONSOLE,
94     "cron",      LOG_CRON,
95     "audit",     LOG_AUDIT,
96     "local0",    LOG_LOCAL0,
97     "local1",    LOG_LOCAL1,
98     "local2",    LOG_LOCAL2,
99     "local3",    LOG_LOCAL3,
100    "local4",    LOG_LOCAL4,
101    "local5",    LOG_LOCAL5,
102    "local6",    LOG_LOCAL6,
103    "local7",    LOG_LOCAL7,
104    NULL,        -1
};

```

unchanged_portion_omitted

```

*****
131413 Mon Nov 11 13:47:11 2013
new/usr/src/cmd/syslogd/syslogd.c
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22  * Copyright (c) 2013 Gary Mills
23  * Copyright 2012 Milan Jurik. All rights reserved.
24  * Copyright 2008 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  * Copyright 2012 Milan Jurik. All rights reserved.
27 */
28 /*
29  * Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T
30  * All Rights Reserved
31 */
32
33 /*
34  * University Copyright- Copyright (c) 1982, 1986, 1988
35  * The Regents of the University of California
36  * All Rights Reserved
37 *
38 * University Acknowledgment- Portions of this document are derived from
39 * software developed by the University of California, Berkeley, and its
40 * contributors.
41 */
42
43 /*
44  * syslogd -- log system messages
45 *
46 * This program implements a system log. It takes a series of lines.
47 * Each line may have a priority, signified as "<n>" as
48 * the first characters of the line. If this is
49 * not present, a default priority is used.
50 *
51 * To kill syslogd, send a signal 15 (terminate). A signal 1 (hup) will
52 * cause it to reconfigure.
53 *
54 * Defined Constants:
55 *
56 * MAXLINE -- the maximum line length that can be handled.
57 * DEFUPRI -- the default priority for user messages.
58 * DEFSPRI -- the default priority for kernel messages.
59 *

```

```

60 */
61
62 #include <unistd.h>
63 #include <note.h>
64 #include <errno.h>
65 #include <sys/types.h>
66 #include <stdio.h>
67 #include <stdio_ext.h>
68 #include <stdlib.h>
69 #include <ctype.h>
70 #include <signal.h>
71 #include <string.h>
72 #include <strings.h>
73 #include <libscf.h>
74 #include <netconfig.h>
75 #include <netdir.h>
76 #include <pwd.h>
77 #include <sys/socket.h>
78 #include <tiuser.h>
79 #include <utmpx.h>
80 #include <limits.h>
81 #include <pthread.h>
82 #include <fcntl.h>
83 #include <stropts.h>
84 #include <assert.h>
85 #include <sys/statvfs.h>
86
87 #include <sys/param.h>
88 #include <sys/sysmacros.h>
89 #include <sys/syslog.h>
90 #include <sys/strlog.h>
91 #include <sys/stat.h>
92 #include <sys/time.h>
93 #include <sys/utsname.h>
94 #include <sys/poll.h>
95 #include <sys/wait.h>
96 #include <sys/resource.h>
97 #include <sys/mman.h>
98 #include <sys/note.h>
99 #include <door.h>
100
101 #include <wchar.h>
102 #include <locale.h>
103 #include <stdarg.h>
104
105 #include "dataq.h"
106 #include "conf.h"
107 #include "syslogd.h"
108
109 #define DOORFILE "/var/run/syslog_door"
110 #define RELATIVE_DOORFILE "../var/run/syslog_door"
111 #define OLD_DOORFILE "/etc/.syslog_door"
112
113 #define PIDFILE "/var/run/syslog.pid"
114 #define RELATIVE_PIDFILE "../var/run/syslog.pid"
115 #define OLD_PIDFILE "/etc/syslog.pid"
116
117 static char *LogName = "/dev/log";
118 static char *ConfFile = "/etc/syslog.conf";
119 static char ctty[] = "/dev/console";
120 static char sysmsg[] = "/dev/sysmsg";
121 static int DoorFd = -1;
122 static int DoorCreated = 0;
123 static int PidfileCreated = 0;
124 static char *DoorFileName = DOORFILE;
125 static char *PidFileName = PIDFILE;

```

```

127 /*
128 * configuration file directives
129 */

131 static struct code      PriNames[] = {
132     "panic",             LOG_EMERG,
133     "emerg",            LOG_EMERG,
134     "alert",            LOG_ALERT,
135     "crit",             LOG_CRIT,
136     "err",              LOG_ERR,
137     "error",            LOG_ERR,
138     "warn",             LOG_WARNING,
139     "warning",          LOG_WARNING,
140     "notice",           LOG_NOTICE,
141     "info",             LOG_INFO,
142     "debug",            LOG_DEBUG,
143     "none",             NOPRI,
144     NULL,               -1
145 };

```

```

147 static struct code      FacNames[] = {
148     "kern",             LOG_KERN,
149     "user",            LOG_USER,
150     "mail",            LOG_MAIL,
151     "daemon",          LOG_DAEMON,
152     "auth",            LOG_AUTH,
153     "security",        LOG_AUTH,
154     "mark",            LOG_MARK,
155     "syslog",          LOG_SYSLOG,
156     "lpr",             LOG_LPR,
157     "news",            LOG_NEWS,
158     "uucp",            LOG_UUCP,
159     "bsdcron",         LOG_BSDCRON,
160     "authpriv",        LOG_AUTHPRIV,
161     "ftp",             LOG_FTP,
162     "ntp",             LOG_NTP,
163     "audit",           LOG_AUDIT,
164     "console",         LOG_CONSOLE,
165     "cron",            LOG_CRON,
166     "local0",          LOG_LOCAL0,
167     "local1",          LOG_LOCAL1,
168     "local2",          LOG_LOCAL2,
169     "local3",          LOG_LOCAL3,
170     "local4",          LOG_LOCAL4,
171     "local5",          LOG_LOCAL5,
172     "local6",          LOG_LOCAL6,
173     "local7",          LOG_LOCAL7,
174     NULL,              -1
175 };

```

unchanged portion omitted

```

606 /*
607 * this thread listens to the local stream log driver for log messages
608 * generated by this host, formats them, and queues them to the logger
609 * thread.
610 */
611 /*ARGSUSED*/
612 static void *
613 sys_poll(void *ap)
614 {
615     int nfd;
616     static int klogerrs = 0;
617     pthread_t mythreadno;

619     if (Debug) {

```

```

620         mythreadno = pthread_self();
621     }

623     DPRINTF(1, "sys_poll(%u): sys_thread started\n", mythreadno);

625     /*
626     * Process messages, blocking on poll because timeout is set
627     * to INFTIM. When poll returns with a message, call getkmsg
628     * to pull up one message from the log driver and enqueue it
629     * with the sync flag set.
630     * Try to process as many messages as we can without blocking on poll.
631     * We count such "initial" messages with sys_init_msg_count and
632     * enqueue them without the SYNC_FILE flag. When no more data is
633     * waiting on the local log device, we set timeout to INFTIM,
634     * clear sys_init_msg_count, and generate a flush message to sync
635     * the previously counted initial messages out to disk.
636     */

632     sys_init_msg_count = 0;

634     for (;;) {
635         errno = 0;
636         t_errno = 0;

638         nfd = poll(&Pfd, 1, INFTIM);

640         if (nfd == 0)
641             continue;

643         if (nfd < 0) {
644             if (errno != EINTR)
645                 logerror("poll");
646             continue;
647         }
648         if (Pfd.revents & POLLIN) {
649             getkmsg(INFTIM);
650         } else {
651             if (shutting_down) {
652                 pthread_exit(0);
653             }
654             if (Pfd.revents & (POLLNVAL|POLLHUP|POLLERR)) {
655                 logerror("kernel log driver poll error");
656                 (void) close(Pfd.fd);
657                 Pfd.fd = -1;
658             }
659         }

661         while (Pfd.fd == -1 && klogerrs++ < 10) {
662             Pfd.fd = openklog(LogName, O_RDONLY);
663         }
664         if (klogerrs >= 10) {
665             logerror("can't reopen kernel log device - fatal");
666             exit(1);
667         }
668     }
669     /*NOTREACHED*/
670     return (NULL);
671 }

673 /*
674 * Pull up one message from log driver.
675 */
676 static void
677 getkmsg(int timeout)
678 {
679     int flags = 0, i;

```

```

680     char *lastline;
681     struct strbuf ctl, dat;
682     struct log_ctl_hdr;
683     char buf[MAXLINE+1];
684     size_t buflen;
685     size_t len;
686     char tmpbuf[MAXLINE+1];
687     pthread_t mythreadno;

689     if (Debug) {
690         mythreadno = pthread_self();
691     }

693     dat.maxlen = MAXLINE;
694     dat.buf = buf;
695     ctl.maxlen = sizeof (struct log_ctl);
696     ctl.buf = (caddr_t)&hdr;

698     while ((i = getmsg(Pfd.fd, &ctl, &dat, &flags)) == MOREDATA) {
699         lastline = &dat.buf[dat.len];
700         *lastline = '\0';

702         DPRINTF2(5, "getkmsg(%u): getmsg: dat.len = %d\n",
698         DPRINTF2(5, "sys_poll:(%u): getmsg: dat.len = %d\n",
703             mythreadno, dat.len);
704         buflen = strlen(buf);
705         len = findnl_bkwd(buf, buflen);

707         (void) memcpy(tmpbuf, buf, len);
708         tmpbuf[len] = '\0';

710         /*
711          * Format sys will enqueue the log message.
712          * Set the sync flag if timeout != 0, which
713          * means that we're done handling all the
714          * initial messages ready during startup.
715          */
716         if (timeout == 0) {
717             formatsys(&hdr, tmpbuf, 0);
718             sys_init_msg_count++;
719         } else {
720             formatsys(&hdr, tmpbuf, 1);
721         }
722         sys_msg_count++;

724         if (len != buflen) {
725             /* If anything remains in buf */
726             size_t remlen;

728             if (buf[len] == '\n') {
729                 /* skip newline */
730                 len++;
731             }

733             /*
734              * Move the remaining bytes to
735              * the beginning of buf.
736              */

738             remlen = buflen - len;
739             (void) memcpy(buf, &buf[len], remlen);
740             dat.maxlen = MAXLINE - remlen;
741             dat.buf = &buf[remlen];
742         } else {
743             dat.maxlen = MAXLINE;
744             dat.buf = buf;

```

```

745     }
746 }

748     if (i == 0 && dat.len > 0) {
749         dat.buf[dat.len] = '\0';
750         /*
751          * Format sys will enqueue the log message.
752          * Set the sync flag if timeout != 0, which
753          * means that we're done handling all the
754          * initial messages ready during startup.
755          */
756         DPRINTF2(5, "getkmsg(%u): getmsg: dat.maxlen = %d\n",
757             mythreadno, dat.maxlen);
758         DPRINTF2(5, "getkmsg(%u): getmsg: dat.len = %d\n",
759             mythreadno, dat.len);
760         DPRINTF2(5, "getkmsg(%u): getmsg: strlen(dat.buf) = %d\n",
761             mythreadno, strlen(dat.buf));
762         DPRINTF2(5, "getkmsg(%u): getmsg: dat.buf = \"%s\"\n",
763             mythreadno, dat.buf);
764         DPRINTF2(5, "getkmsg(%u): buf len = %d\n",
765             mythreadno, strlen(buf));
766         if (timeout == 0) {
767             formatsys(&hdr, buf, 0);
768             sys_init_msg_count++;
769         } else {
770             formatsys(&hdr, buf, 1);
771         }
772         sys_msg_count++;
773     } else if (i < 0 && errno != EINTR) {
774         if (!shutting_down) {
775             logerror("kernel log driver read error");
776         }
777         (void) close(Pfd.fd);
778         Pfd.fd = -1;
779     }
780 }

```

unchanged portion omitted

```

*****
10215 Mon Nov 11 13:47:11 2013
new/usr/src/man/man1m/syslogd.1m
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1 \" te
2 .\" Copyright (c) 2013 Gary Mills
3 .\" Copyright (C) 2008, Sun Microsystems, Inc. All Rights Reserved
4 .\" The contents of this file are subject to the terms of the Common Development
5 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
6 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
7 .TH SYSLOGD 1M \"Oct 17, 2013\"
8 .TH SYSLOGD 1M \"Oct 31, 2008\"
9 .SH NAME
10 syslogd \- log system messages
11 .SH SYNOPSIS
12 .LP
13 \fB/usr/sbin/syslogd\fR [\fB-d\fR] [\fB-f\fR \fIconfigfile\fR] [\fB-m\fR \fImark
14 [\fB-p\fR \fIpath\fR] [\fB-t\fR | \fB-T\fR]
15 .fi

17 .SH DESCRIPTION
18 .sp
19 .LP
20 \fBsyslogd\fR reads and forwards system messages to the appropriate log files
21 or users, depending upon the priority of a message and the system facility from
22 which it originates. The configuration file \fB/etc/syslog.conf\fR (see
23 \fBsyslog.conf\fR(4)) controls where messages are forwarded. \fBsyslogd\fR logs
24 a mark (timestamp) message every \fImarkinterval\fR minutes (default \fB20\fR)
25 at priority \fBLOG_INFO\fR to the facility whose name is given as \fBmark\fR in
26 the \fBsyslog.conf\fR file.
27 .sp
28 .LP
29 A system message consists of a single line of text, which may be prefixed with
30 a priority code number enclosed in angle-brackets (\fB<|>\fR); priorities are
31 defined in \fB<sys/syslog.h>\fR&.
32 .sp
33 .LP
34 \fBsyslogd\fR reads from the \fBSTREAMS\fR log driver, \fB/dev/log\fR, and from
35 any transport provider specified in \fB/etc/netconfig\fR,
36 \fB/etc/net/transport/hosts\fR, and \fB/etc/net/transport/services\fR.
37 .sp
38 .LP
39 \fBsyslogd\fR reads the configuration file when it starts up, and again
40 whenever it receives a \fBHUP\fR signal (see \fBsignal.h\fR(3HEAD)), at which
41 time it also closes all files it has open, re-reads its configuration file, and
42 then opens only the log files that are listed in that file. \fBsyslogd\fR exits
43 when it receives a \fBTERM\fR signal.
44 .sp
45 .LP
46 As it starts up, \fBsyslogd\fR creates the file \fB/var/run/syslog.pid\fR, if
47 possible, containing its process identifier (\fBPID\fR).
48 .sp
49 .LP
50 If message \fBID\fR generation is enabled (see \fBlog\fR(7D)), each message
51 will be preceded by an identifier in the following format: \fB[ID\fR \fImsgid
52 facility\fR\fB&.\fR\fIpriority\fR\fB]\fR. \fImsgid\fR is the message's numeric
53 identifier described in \fBmsgid\fR(1M). \fIfacility\fR and \fIpriority\fR are
54 described in \fBsyslog.conf\fR(4). \fB[ID 123456 kern.notice]\fR is an example
55 of an identifier when message \fBID\fR generation is enabled.
56 .sp
57 .LP
58 If the message originated in a loadable kernel module or driver, the kernel
59 module's name (for example, \fBbufs\fR) will be displayed instead of \fBunix\fR.

```

```

60 See \fBEXAMPLES\fR for sample output from \fBsyslogd\fR with and without
61 message \fBID\fR generation enabled.
62 .sp
63 .LP
64 In an effort to reduce visual clutter, message \fBID\fRs are not displayed when
65 writing to the console; message \fBID\fRs are only written to the log file.
66 See \fBEXAMPLES\fR.
67 .sp
68 .LP
69 The \fB/etc/default/syslogd\fR file contains the following default parameter
70 settings, which are in effect if neither the \fB-t\fR nor \fB-T\fR option is
71 selected. See \fBFILES\fR.
72 .sp
73 .LP
74 The recommended way to allow or disallow message logging is through the use of
75 the service management facility (\fBsmf\fR(5)) property:
76 .sp
77 .in +2
78 .nf
79 svc:/system/system-log/config/log_from_remote
80 .fi
81 .in -2

83 .sp
84 .LP
85 This property specifies whether remote messages are logged.
86 \fBlog_from_remote=true\fR is equivalent to the \fB-t\fR command-line option
87 and \fBfalse\fR is equivalent to the \fB-T\fR command-line option. The default
88 value for \fBlog_from_remote\fR is \fBfalse\fR. See NOTES, below.
89 .sp
90 .ne 2
91 .na
92 \fB\fblog_from_remote\fR
93 .ad
94 .sp .6
95 .RS 4n
96 Specifies whether remote messages are logged. \fBLOG_FROM_REMOTE=NO\fR is
97 equivalent to the \fB-t\fR command-line option. The default value for
98 \fBLOG_FROM_REMOTE\fR is \fBYES\fR.
99 .RE

101 .SH OPTIONS
102 .sp
103 .LP
104 The following options are supported:
105 .sp
106 .ne 2
107 .na
108 \fB-d\fR
109 .ad
110 .sp .6
111 .RS 4n
112 Turn on debugging. This option should only be used interactively in a root
113 shell once the system is in multi-user mode. It should \fBnot\fR be used in the
114 system start-up scripts, as this will cause the system to hang at the point
115 where \fBsyslogd\fR is started.
116 .RE

118 .sp
119 .ne 2
120 .na
121 \fB-f\fR \fIconfigfile\fR
122 .ad
123 .sp .6
124 .RS 4n

```

```

125 Specify an alternate configuration file.
126 .RE

128 .sp
129 .ne 2
130 .na
131 \fB\fB-m\fR \fImarkinterval\fR\fR
132 .ad
133 .sp .6
134 .RS 4n
135 Specify an interval, in minutes, between mark messages.
136 .RE

138 .sp
139 .ne 2
140 .na
141 \fB\fB-p\fR \fIpath\fR\fR
142 .ad
143 .sp .6
144 .RS 4n
145 Specify an alternative log device name. The default is \fB/dev/log\fR.
146 .RE

148 .sp
149 .ne 2
150 .na
151 \fB\fB-T\fR\fR
152 .ad
153 .sp .6
154 .RS 4n
155 Enable the \fBsyslogd\fR \fBUDP\fR port to turn on logging of remote messages.
156 This is the default behavior. See \fBEXAMPLES\fR.
157 This is the default behavior. See .
158 .RE

159 .sp
160 .ne 2
161 .na
162 \fB\fB-t\fR\fR
163 .ad
164 .sp .6
165 .RS 4n
166 Disable the \fBsyslogd\fR \fBUDP\fR port to turn off logging of remote
167 messages. See \fBEXAMPLES\fR.
168 messages. See .
169 .RE

170 .SH EXAMPLES
171 .LP
172 \fBExample 1 \fR\fBsyslogd\fR Output Without Message ID Generation Enabled
173 .sp
174 .LP
175 The following example shows the output from \fBsyslogd\fR when message \fBID\fR
176 generation \fBis not\fR enabled:

178 .sp
179 .in +2
180 .nf
181 Sep 29 21:41:18 cathy unix: alloc /: file system full
182 .fi
183 .in -2
184 .sp

186 .LP
187 \fBExample 2 \fR\fBsyslogd\fR Output with ID generation Enabled
188 .sp

```

```

189 .LP
190 The following example shows the output from \fBsyslogd\fR when message \fBID\fR
191 generation \fBis\fR enabled. The message \fBID\fR is displayed when writing to
192 log file \fB/var/adm/messages\fR.

194 .sp
195 .in +2
196 .nf
197 Sep 29 21:41:18 cathy ufs: [ID 845546 kern.notice]
198                                     alloc /: file system full
199 .fi
200 .in -2
201 .sp

203 .LP
204 \fBExample 3 \fR\fBsyslogd\fR Output with ID Generation Enabled
205 .sp
206 .LP
207 The following example shows the output from \fBsyslogd\fR when message \fBID\fR
208 generation \fBis\fR enabled when writing to the console. Even though message ID
209 is enabled, the message \fBID\fR is not displayed at the console.

211 .sp
212 .in +2
213 .nf
214 Sep 29 21:41:18 cathy ufs: alloc /: file system full
215 .fi
216 .in -2
217 .sp

219 .LP
220 \fBExample 4 \fREnabling Acceptance of UDP Messages from Remote Systems
221 .sp
222 .LP
223 The following commands enable \fBsyslogd\fR to accept entries from remote
224 systems.

226 .sp
227 .in +2
228 .nf
229 # \fBsvccfg -s svc:/system/system-log setprop config/log_from_remote = true\fR
230 # \fBsvcadm restart svc:/system/system-log\fR
231 .fi
232 .in -2
233 .sp

235 .SH FILES
236 .sp
237 .ne 2
238 .na
239 \fB\fB/etc/syslog.conf\fR\fR
240 .ad
241 .sp .6
242 .RS 4n
243 Configuration file
244 .RE

246 .sp
247 .ne 2
248 .na
249 \fB\fB/var/run/syslog.pid\fR\fR
250 .ad
251 .sp .6
252 .RS 4n
253 Process \fBID\fR
254 .RE

```

```

256 .sp
257 .ne 2
258 .na
259 \fB\fB/etc/default/syslogd\fR\fR
260 .ad
261 .sp .6
262 .RS 4n
263 Contains default settings. You can override some of the settings by
264 command-line options.
265 .RE

267 .sp
268 .ne 2
269 .na
270 \fB\fB/dev/log\fR\fR
271 .ad
272 .sp .6
273 .RS 4n
274 \fBFBSTREAMS\fR log driver
275 .RE

277 .sp
278 .ne 2
279 .na
280 \fB\fB/etc/netconfig\fR\fR
281 .ad
282 .sp .6
283 .RS 4n
284 Transport providers available on the system
285 .RE

287 .sp
288 .ne 2
289 .na
290 \fB\fB/etc/net/transport/hosts\fR\fR
291 .ad
292 .sp .6
293 .RS 4n
294 Network hosts for each transport
295 .RE

297 .sp
298 .ne 2
299 .na
300 \fB\fB/etc/net/transport/services\fR\fR
301 .ad
302 .sp .6
303 .RS 4n
304 Network services for each transport
305 .RE

307 .SH SEE ALSO
308 .sp
309 .LP
310 \fBfBlogger\fR(1), \fBfBsvcs\fR(1), \fBfBmsgid\fR(1M), \fBfBsvcadm\fR(1M),
311 \fBfBsvccfg\fR(1M), \fBfBsyslog\fR(3C), \fBfBsyslog.conf\fR(4), \fBfBattributes\fR(5),
312 \fBfBsignal.h\fR(3HEAD), \fBfBsmf\fR(5), \fBfBlog\fR(7D)
313 .SH NOTES
314 .sp
315 .LP
316 The \fBfBmark\fR message is a system time stamp, and so it is only defined for
317 the system on which \fBfBsyslogd\fR is running. It can not be forwarded to other
318 systems.
319 .sp
320 .LP

```

```

321 When \fBfBsyslogd\fR receives a \fBfBHUP\fR signal, it attempts to complete
322 outputting pending messages, and close all log files to which it is currently
323 logging messages. If, for some reason, one (or more) of these files does not
324 close within a generous grace period, \fBfBsyslogd\fR discards the pending
325 messages, forcibly closes these files, and starts reconfiguration. If this
326 shutdown procedure is disturbed by an unexpected error and \fBfBsyslogd\fR cannot
327 complete reconfiguration, \fBfBsyslogd\fR sends a mail message to the superuser
328 on the current system stating that it has shut down, and exits.
329 .sp
330 .LP
331 Care should be taken to ensure that each window displaying messages forwarded
332 by \fBfBsyslogd\fR (especially console windows) is run in the system default
333 locale (which is \fBfBsyslogd\fR's locale). If this advice is not followed, it is
334 possible for a \fBfBsyslogd\fR message to alter the terminal settings for that
335 window, possibly even allowing remote execution of arbitrary commands from that
336 window.
337 .sp
338 .LP
339 The \fBfBsyslogd\fR service is managed by the service management facility,
340 \fBfBfBsmf\fR(5), under the service identifier:
341 .sp
342 .in +2
343 .nf
344 svc:/system/system-log:default
345 .fi
346 .in -2
347 .sp

349 .sp
350 .LP
351 Administrative actions on this service, such as enabling, disabling, or
352 requesting restart, can be performed using \fBfBsvcadm\fR(1M). The service's
353 status can be queried using the \fBfBsvcs\fR(1) command.
354 .sp
355 .LP
356 When \fBfBsyslogd\fR is started by means of \fBfBsvcadm\fR(1M), if a value is
357 specified for \fBfBLOG_FROM_REMOTE\fR in the \fBfB/etc/defaults/syslogd\fR file,
358 the SMF property \fBfBsvc:/system/system-log/config/log_from_remote\fR is set to
359 correspond to the \fBfBLOG_FROM_REMOTE\fR value and the
360 \fBfB/etc/default/syslogd\fR file is modified to replace the
361 \fBfBLOG_FROM_REMOTE\fR specification with the following comment:
362 .sp
363 .in +2
364 .nf
365 # LOG_FROM_REMOTE is now set using svccfg(1m), see syslogd(1m).
366 .fi
367 .in -2

369 .sp
370 .LP
371 If neither \fBfBLOG_FROM_REMOTE\fR nor
372 \fBfBsvc:/system/system-log/config/log_from_remote\fR are defined, the default is
373 to log remote messages.
374 .sp
375 .LP
376 On installation, the initial value of
377 \fBfBsvc:/system/system-log/config/log_from_remote\fR is \fBfBfalse\fR.

```

```

*****
      8168 Mon Nov 11 13:47:11 2013
new/usr/src/man/man4/syslog.conf.4
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1  \" te
2  .\" Copyright (c) 2013 Gary Mills
3  .\" Copyright (c) 2003 Sun Microsystems, Inc. All Rights Reserved.
4  .\" Copyright (c) 1983 Regents of the University of California. All rights rese
5  .TH SYSLOG.CONF 4 "Oct 17, 2013"
4  .TH SYSLOG.CONF 4 "Apr 26, 2006"
6  .SH NAME
7  syslog.conf \- configuration file for syslogd system log daemon
8  .SH SYNOPSIS
9  .LP
10 .nf
11 \fB/etc/syslog.conf\fR
12 .fi

14 .SH DESCRIPTION
15 .sp
16 .LP
17 The file \fB/etc/syslog.conf\fR contains information used by the system log
18 daemon, \fBsyslogd\fR(1M), to forward a system message to appropriate log files
19 and/or users. \fBsyslogd\fR preprocesses this file through \fBm4\fR(1) to
20 obtain the correct information for certain log files, defining \fBLOGHOST\fR if
21 the address of "loghost" is the same as one of the addresses of the host that
22 is running \fBsyslogd\fR.
23 .sp
24 .LP
25 A configuration entry is composed of two TAB-separated fields:
26 .sp
27 .in +2
28 .nf
29 \fIselector          action\fR
30 .fi
31 .in -2

33 .sp
34 .LP
35 The \fIselector\fR field contains a semicolon-separated list of priority
36 specifications of the form:
37 .sp
38 .in +2
39 .nf
40 \fIfacility\fR\fB&.\fR\fIlevel\fR [ \fB;\fR \fIfacility\fR\fB&.\fR\fIlevel\fR
41 .fi
42 .in -2

44 .sp
45 .LP
46 where \fIfacility\fR is a system facility, or comma-separated list of
47 facilities, and \fIlevel\fR is an indication of the severity of the condition
48 being logged. Recognized values for \fIfacility\fR include:
49 .sp
50 .ne 2
51 .na
52 \fB\bkern\fR
51 \fB\buser\fR
53 .ad
54 .RS 12n
55 Messages generated by the kernel.
54 Messages generated by user processes. This is the default priority for messages
55 from programs or facilities not listed in this file.
56 .RE

```

```

58 .sp
59 .ne 2
60 .na
61 \fB\buser\fR
61 \fB\bkern\fR
62 .ad
63 .RS 12n
64 Messages generated by user processes. This is the default priority for messages
65 from programs or facilities not listed in this file.
64 Messages generated by the kernel.
66 .RE

68 .sp
69 .ne 2
70 .na
71 \fB\bmail\fR
72 .ad
73 .RS 12n
74 The mail system.
75 .RE

77 .sp
78 .ne 2
79 .na
80 \fB\bdaemon\fR
81 .ad
82 .RS 12n
83 System daemons, such as \fBbin.ftpd\fR(1M)
84 .RE

86 .sp
87 .ne 2
88 .na
89 \fB\bauth\fR
90 .ad
91 .RS 12n
92 The authorization system: \fBlogin\fR(1), \fBbsu\fR(1M), \fBbgetty\fR(1M), among
93 others.
94 .RE

96 .sp
97 .ne 2
98 .na
99 \fB\bldpr\fR
100 .ad
101 .RS 12n
102 The line printer spooling system: \fBldpr\fR(1B), \fBldpc\fR(1B), among others.
103 .RE

105 .sp
106 .ne 2
107 .na
108 \fB\bnews\fR
109 .ad
110 .RS 12n
111 Designated for the USENET network news system.
112 .RE

114 .sp
115 .ne 2
116 .na
117 \fB\bucp\fR
118 .ad
119 .RS 12n
120 Designated for the UUCP system; it does not currently use the \fBsyslog\fR

```



```

121 mechanism.
122 .RE

124 .sp
125 .ne 2
126 .na
127 \fB\fBbsdcron\fR\fR
126 \fB\fBcron\fR\fR
128 .ad
129 .RS 12n
130 Designated for the BSD cron/at system.
129 Designated for \fBcron\fR/\fBat\fR messages generated by systems that do
130 logging through \fBsyslog\fR. The current version of the Solaris Operating
131 Environment does not use this facility for logging.
131 .RE

133 .sp
134 .ne 2
135 .na
136 \fB\fBauthpriv\fR\fR
137 .ad
138 .RS 12n
139 Designated for the BSD security/authorization system.
140 .RE

142 .sp
143 .ne 2
144 .na
145 \fB\fBftp\fR\fR
146 .ad
147 .RS 12n
148 Designated for the file transfer system.
149 .RE

151 .sp
152 .ne 2
153 .na
154 \fB\fBntp\fR\fR
155 .ad
156 .RS 12n
157 Designated for the network time system.
158 .RE

160 .sp
161 .ne 2
162 .na
163 \fB\fBaudit\fR\fR
164 .ad
165 .RS 12n
166 Designated for audit messages generated by systems that audit by means of
167 syslog.
168 .RE

170 .sp
171 .ne 2
172 .na
173 \fB\fBconsole\fR\fR
174 .ad
175 .RS 12n
176 Designated for the BSD console system.
177 .RE

179 .sp
180 .ne 2
181 .na
182 \fB\fBcron\fR\fR

```

```

183 .ad
184 .RS 12n
185 Designated for \fBcron\fR/\fBat\fR messages generated by systems that do
186 logging through \fBsyslog\fR. The current version of the Solaris Operating
187 Environment does not use this facility for logging.
188 .RE

190 .sp
191 .ne 2
192 .na
193 \fB\fBlocal0-7\fR\fR
194 .ad
195 .RS 12n
196 Designated for local use.
197 .RE

199 .sp
200 .ne 2
201 .na
202 \fB\fBmark\fR\fR
203 .ad
204 .RS 12n
205 For timestamp messages produced internally by \fBsyslogd\fR.
206 .RE

208 .sp
209 .ne 2
210 .na
211 \fB\fB*\fR\fR
212 .ad
213 .RS 12n
214 An asterisk indicates all facilities except for the \fBmark\fR facility.
215 .RE

217 .sp
218 .LP
219 Recognized values for \fIlevel\fR are (in descending order of severity):
220 .sp
221 .ne 2
222 .na
223 \fB\fBemerg\fR\fR
224 .ad
225 .RS 11n
226 For panic conditions that would normally be broadcast to all users.
227 .RE

229 .sp
230 .ne 2
231 .na
232 \fB\fBalert\fR\fR
233 .ad
234 .RS 11n
235 For conditions that should be corrected immediately, such as a corrupted system
236 database.
237 .RE

239 .sp
240 .ne 2
241 .na
242 \fB\fBcrit\fR\fR
243 .ad
244 .RS 11n
245 For warnings about critical conditions, such as hard device errors.
246 .RE

248 .sp

```

```

249 .ne 2
250 .na
251 \fB\fBerr\fR\fR
252 .ad
253 .RS 11n
254 For other errors.
255 .RE

257 .sp
258 .ne 2
259 .na
260 \fB\fBwarning\fR\fR
261 .ad
262 .RS 11n
263 For warning messages.
264 .RE

266 .sp
267 .ne 2
268 .na
269 \fB\fBnotice\fR\fR
270 .ad
271 .RS 11n
272 For conditions that are not error conditions, but may require special handling.
273 A configuration entry with a \fIlevel\fR value of \fBnotice\fR must appear on a
274 separate line.
275 .RE

277 .sp
278 .ne 2
279 .na
280 \fB\fBinfo\fR\fR
281 .ad
282 .RS 11n
283 Informational messages.
284 .RE

286 .sp
287 .ne 2
288 .na
289 \fB\fBdebug\fR\fR
290 .ad
291 .RS 11n
292 For messages that are normally used only when debugging a program.
293 .RE

295 .sp
296 .ne 2
297 .na
298 \fB\fBnone\fR\fR
299 .ad
300 .RS 11n
301 Do not send messages from the indicated \fIfacility\fR to the selected file.
302 For example, a \fIselector\fR of
303 .sp
304 \fB*.debug/mail.none\fR
305 .sp
306 sends all messages \fIexcept\fR mail messages to the selected file.
307 .RE

309 .sp
310 .LP
311 For a given \fIfacility\fR and \fIlevel\fR, \fBsyslogd\fR matches all messages
312 for that level and all higher levels. For example, an entry that specifies a
313 level of \fBcrit\fR also logs messages at the \fBalert\fR and \fBemerg\fR
314 levels.

```

```

315 .sp
316 .LP
317 The \fIfaction\fR field indicates where to forward the message. Values for this
318 field can have one of four forms:
319 .RS +4
320 .TP
321 .ie t \(\bu
322 .el o
323 A filename, beginning with a leading slash, which indicates that messages
324 specified by the \fIselector\fR are to be written to the specified file. The
325 file is opened in append mode if it exists. If the file does not exist, logging
326 silently fails for this action.
327 .RE
328 .RS +4
329 .TP
330 .ie t \(\bu
331 .el o
332 The name of a remote host, prefixed with an \fB@\fR, as with:
333 \fB@\fR\fIserver\fR, which indicates that messages specified by the
334 \fIselector\fR are to be forwarded to the \fBsyslogd\fR on the named host. The
335 hostname "loghost" is treated, in the default \fBsyslog.conf\fR, as the
336 hostname given to the machine that logs \fBsyslogd\fR messages. Every machine
337 is "loghost" by default, per the hosts database. It is also possible to specify
338 one machine on a network to be "loghost" by, literally, naming the machine
339 "loghost". If the local machine is designated to be "loghost", then
340 \fBsyslogd\fR messages are written to the appropriate files. Otherwise, they
341 are sent to the machine "loghost" on the network.
342 .RE
343 .RS +4
344 .TP
345 .ie t \(\bu
346 .el o
347 A comma-separated list of usernames, which indicates that messages specified by
348 the \fIselector\fR are to be written to the named users if they are logged in.
349 .RE
350 .RS +4
351 .TP
352 .ie t \(\bu
353 .el o
354 An asterisk, which indicates that messages specified by the \fIselector\fR are
355 to be written to all logged-in users.
356 .RE
357 .sp
358 .LP
359 Blank lines are ignored. Lines for which the first nonwhite character is
360 a '\fB#\fR' are treated as comments.
361 .SH EXAMPLES
362 .LP
363 \fBExample 1 \fR Sample Configuration File
364 .sp
365 .LP
366 With the following configuration file:
367 .sp
368 .sp
370 .sp
371 .TS
372 l l
373 l l .
374 \fB*.notice\fR \fB/var/log/notice\fR
375 \fBmail.info\fR \fB/var/log/notice\fR
376 \fB*.crit\fR \fB/var/log/critical\fR
377 \fBkern,mark.debug\fR \fB/dev/console\fR
378 \fBkern.err\fR \fB@server\fR
379 \fB*.emerg\fR \fB*\fR
380 \fB*.alert\fR \fBroot,operator\fR

```

```
381 \fB*.alert;auth.warning\fR      \fB/var/log/auth\fR
382 .TE

384 .sp
385 .LP
386 \fBsyslogd\fR(1M) logs all mail system messages except \fBdebug\fR messages and
387 all \fBnotice\fR (or higher) messages into a file named \fB/var/log/notice\fR.
388 It logs all critical messages into \fB/var/log/critical\fR, and all kernel
389 messages and 20-minute marks onto the system console.

391 .sp
392 .LP
393 Kernel messages of \fBerr\fR (error) severity or higher are forwarded to the
394 machine named \fBserver\fR. Emergency messages are forwarded to all users. The
395 users \fBroot\fR and \fBoperator\fR are informed of any \fBalert\fR messages.
396 All messages from the authorization system of \fBwarning\fR level or higher are
397 logged in the file \fB/var/log/auth\fR.

399 .SH ATTRIBUTES
400 .sp
401 .LP
402 See \fBattributes\fR(5) for descriptions of the following attributes:
403 .sp

405 .sp
406 .TS
407 box:
408 c | c
409 l | l .
410 ATTRIBUTE TYPE      ATTRIBUTE VALUE
411 -
412 Interface Stability      Stable
413 .TE

415 .SH SEE ALSO
416 .sp
417 .LP
418 \fBat\fR(1), \fBcrontab\fR(1), \fBlogger\fR(1), \fBlogin\fR(1), \fBlp\fR(1),
419 \fBlpc\fR(1B), \fBlpr\fR(1B), \fBm4\fR(1), \fBcron\fR(1M), \fBgetty\fR(1M),
420 \fBin.ftpd\fR(1M), \fBsu\fR(1M), \fBsyslogd\fR(1M), \fBsyslog\fR(3C),
421 \fBhosts\fR(4), \fBattributes\fR(5)
```

```

*****
19190 Mon Nov 11 13:47:11 2013
new/usr/src/uts/common/os/logsubr.c
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */

22 /*
23 * Copyright (c) 2013 Gary Mills
24 * Copyright (c) 1998, 2010, Oracle and/or its affiliates. All rights reserved.
25 */

27 #include <sys/types.h>
28 #include <sys/param.h>
29 #include <sys/varargs.h>
30 #include <sys/system.h>
31 #include <sys/cmn_err.h>
32 #include <sys/stream.h>
33 #include <sys/strsubr.h>
34 #include <sys/strsun.h>
35 #include <sys/sysmacros.h>
36 #include <sys/kmem.h>
37 #include <sys/log.h>
38 #include <sys/spl.h>
39 #include <sys/syslog.h>
40 #include <sys/console.h>
41 #include <sys/debug.h>
42 #include <sys/utsname.h>
43 #include <sys/id_space.h>
44 #include <sys/zone.h>

46 log_zone_t log_global;
47 queue_t *log_consq;
48 queue_t *log_backlogq;
49 queue_t *log_intrq;

51 #define LOG_PRI_SIZE 8 /* max priority size: 7 characters + null */
52 #define LOG_FAC_SIZE 9 /* max priority size: 8 characters + null */

54 static krwlock_t log_rwlock;
55 static int log_rwlock_depth;
56 static int log_seq_no[SL_CONSOLE + 1];
57 static stdata_t log_fakestr;
58 static id_space_t *log_minorspace;
59 static log_t log_backlog;
60 static struct kmem_cache *log_cons_cache; /* log_t cache */

```

```

62 static queue_t *log_recentq;
63 static queue_t *log_freeq;

65 static zone_key_t log_zone_key;

67 static char log_overflow_msg[] = "message overflow on /dev/log minor %#d%s\n";

69 static char log_pri[LOG_PRIMASK + 1][LOG_PRI_SIZE] = {
70     "emerg",      "alert",      "crit",      "error",
71     "warning",    "notice",    "info",      "debug"
72 };

74 static char log_fac[LOG_NFACILITIES + 1][LOG_FAC_SIZE] = {
75     "kern",      "user",      "mail",      "daemon",
76     "auth",      "syslog",    "lpr",      "news",
77     "uucp",      "bsdcron",   "authpriv",  "ftp",
78     "ntp",        "audit",     "console",   "cron",
79     "uucp",      "resv9",     "resv10",    "resv11",
80     "resv12",    "audit",     "resv14",    "cron",
81     "local0",    "local1",    "local2",    "local3",
82     "local4",    "local5",    "local6",    "local7",
83     "unknown"
84 };

```

unchanged_portion_omitted

new/usr/src/uts/common/sys/syslog.h

1

```
*****
3229 Mon Nov 11 13:47:12 2013
new/usr/src/uts/common/sys/syslog.h
4211 Some syslog facility names and symbols are missing
1762 Syslogd man page: missing reference.
*****
1 /*
2  * Copyright (c) 2013 Gary Mills
3  * Copyright 2003 Sun Microsystems, Inc. All rights reserved.
4  * Use is subject to license terms.
5  */

7 /*      Copyright (c) 1988 AT&T */
8 /*      All Rights Reserved */

10 /*
11  * University Copyright- Copyright (c) 1982, 1986, 1988
12  * The Regents of the University of California
13  * All Rights Reserved
14  *
15  * University Acknowledgment- Portions of this document are derived from
16  * software developed by the University of California, Berkeley, and its
17  * contributors.
18  */

20 #ifndef _SYS_SYSLOG_H
21 #define _SYS_SYSLOG_H

22 #pragma ident      "%Z%M% %I%      %E% SMI"

23 #ifdef __cplusplus
24 extern "C" {
25 #endif

27 /*
28  * Facility codes
29  */
30 #define LOG_KERN      (0<<3) /* kernel messages */
31 #define LOG_USER      (1<<3) /* random user-level messages */
32 #define LOG_MAIL      (2<<3) /* mail system */
33 #define LOG_DAEMON    (3<<3) /* system daemons */
34 #define LOG_AUTH      (4<<3) /* security/authorization messages */
35 #define LOG_SYSLOG    (5<<3) /* messages generated internally by syslogd */
36 #define LOG_LPR       (6<<3) /* line printer subsystem */
37 #define LOG_NEWS      (7<<3) /* netnews subsystem */
38 #define LOG_UUCP      (8<<3) /* uucp subsystem */
39 #define LOG_BSDCRON   (9<<3) /* BSD cron/at subsystem */
40 #define LOG_AUTHPRIV  (10<<3) /* BSD security/authorization messages */
41 #define LOG_FTP       (11<<3) /* file transfer subsystem */
42 #define LOG_NTP       (12<<3) /* network time subsystem */
43 #define LOG_AUDIT     (13<<3) /* audit subsystem */
44 #define LOG_CONSOLE   (14<<3) /* BSD console messages */
45 #define LOG_CRON      (15<<3) /* cron/at subsystem */
46 #define LOG_LOCAL0    (16<<3) /* reserved for local use */
47 #define LOG_LOCAL1    (17<<3) /* reserved for local use */
48 #define LOG_LOCAL2    (18<<3) /* reserved for local use */
49 #define LOG_LOCAL3    (19<<3) /* reserved for local use */
50 #define LOG_LOCAL4    (20<<3) /* reserved for local use */
51 #define LOG_LOCAL5    (21<<3) /* reserved for local use */
52 #define LOG_LOCAL6    (22<<3) /* reserved for local use */
53 #define LOG_LOCAL7    (23<<3) /* reserved for local use */

55 #define LOG_NFACILITIES 24 /* maximum number of facilities */
56 #define LOG_FACMASK    0x03f8 /* mask to extract facility part */
```

new/usr/src/uts/common/sys/syslog.h

2

```
58 /*
59  * Priorities (these are ordered)
60  */
61 #define LOG_EMERG      0 /* system is unusable */
62 #define LOG_ALERT      1 /* action must be taken immediately */
63 #define LOG_CRIT       2 /* critical conditions */
64 #define LOG_ERR        3 /* error conditions */
65 #define LOG_WARNING    4 /* warning conditions */
66 #define LOG_NOTICE     5 /* normal but signification condition */
67 #define LOG_INFO       6 /* informational */
68 #define LOG_DEBUG      7 /* debug-level messages */

70 #define LOG_PRIMASK    0x0007 /* mask to extract priority part (internal) */

72 /*
73  * arguments to setlogmask.
74  */
75 #define LOG_MASK(pri) (1 << (pri)) /* mask for one priority */
76 #define LOG_UPTO(pri) ((1 << ((pri)+1)) - 1) /* all priorities through pri */

78 /*
79  * Option flags for openlog.
80  */
81 * LOG_ODELAY no longer does anything; LOG_NDELAY is the
82 * inverse of what it used to be.
83 */
84 #define LOG_PID        0x01 /* log the pid with each message */
85 #define LOG_CONS       0x02 /* log on the console if errors in sending */
86 #define LOG_ODELAY     0x04 /* delay open until syslog() is called */
87 #define LOG_NDELAY     0x08 /* don't delay open */
88 #define LOG_NOWAIT     0x10 /* if forking to log on console, don't wait() */

90 #ifdef __cplusplus
91 }

```

unchanged portion omitted