```
**********************************************************
    7903 Mon Oct 28 12:46:17 2013
new/usr/src/cmd/logger/logger.c
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
1762 Syslogd man page: missing reference.
**********************************************************
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright (c) 2013 Gary Mills
 * Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

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

/*
 * University Copyright- Copyright (c) 1982, 1986, 1988
 * The Regents of the University of California
 * All Rights Reserved
 *
 * University Acknowledgment- Portions of this document are derived from
 * software developed by the University of California, Berkeley, and its
 * contributors.
 */

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

#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <syslog.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>
#include <locale.h>
#include <limits.h>
#include <pwd.h>
#include <errno.h>

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

struct code {
        char    *c_name;
        int     c_val;
```



```
};
______unchanged_portion_omitted_

static struct code      FacNames[] = {
        "kern",         LOG_KERN,
        "user",         LOG_USER,
        "mail",         LOG_MAIL,
        "daemon",       LOG_DAEMON,
        "auth",         LOG_AUTH,
        "security",     LOG_AUTH,
        "mark",         LOG_MARK,
        "syslog",       LOG_SYSLOG,
        "lpr",          LOG_LPR,
        "news",         LOG_NEWS,
        "uucp",         LOG_UUCP,
        "bsdcron",      LOG_BSDCRON,
        "authpriv",     LOG_AUTHPRIV,
        "ftp",          LOG_FTP,
        "ntp",          LOG_NTP,
        "audit",        LOG_AUDIT,
        "console",      LOG_CONSOLE,
        "cron",         LOG_CRON,
        "audit",        LOG_AUDIT,
        "local0",       LOG_LOCAL0,
        "local1",       LOG_LOCAL1,
        "local2",       LOG_LOCAL2,
        "local3",       LOG_LOCAL3,
        "local4",       LOG_LOCAL4,
        "local5",       LOG_LOCAL5,
        "local6",       LOG_LOCAL6,
        "local7",       LOG_LOCAL7,
        NULL,           -1
};
______unchanged_portion_omitted_
```

```
**********************************************************
  131681 Mon Oct 28 12:46:17 2013
new/usr/src/cmd/syslogd/syslogd.c
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
1762 Syslogd man page: missing reference.
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright (c) 2013 Gary Mills
 * Copyright 2012 Milan Jurik. All rights reserved.
 * Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 * Copyright 2012 Milan Jurik. All rights reserved.
 */

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

/*
 * University Copyright- Copyright (c) 1982, 1986, 1988
 * The Regents of the University of California
 * All Rights Reserved
 *
 * University Acknowledgment- Portions of this document are derived from
 * software developed by the University of California, Berkeley, and its
 * contributors.
 */

/*
 *  syslogd -- log system messages
 *
 * This program implements a system log. It takes a series of lines.
 * Each line may have a priority, signified as "<n>" as
 * the first characters of the line.  If this is
 * not present, a default priority is used.
 *
 * To kill syslogd, send a signal 15 (terminate).  A signal 1 (hup) will
 * cause it to reconfigure.
 *
 * Defined Constants:
 *
 * MAXLINE -- the maximimum line length that can be handled.
 * DEFUPRI -- the default priority for user messages.
 * DEFSPRI -- the default priority for kernel messages.
```



```
 *
 */

#include <unistd.h>
#include <note.h>
#include <errno.h>
#include <sys/types.h>
#include <stdio.h>
#include <stdio_ext.h>
#include <stdlib.h>
#include <ctype.h>
#include <signal.h>
#include <string.h>
#include <strings.h>
#include <libscf.h>
#include <netconfig.h>
#include <netdir.h>
#include <pwd.h>
#include <sys/socket.h>
#include <tiuser.h>
#include <utmpx.h>
#include <limits.h>
#include <pthread.h>
#include <fcntl.h>
#include <stropts.h>
#include <assert.h>
#include <sys/statvfs.h>

#include <sys/param.h>
#include <sys/sysmacros.h>
#include <sys/syslog.h>
#include <sys/strlog.h>
#include <sys/stat.h>
#include <sys/time.h>
#include <sys/utsname.h>
#include <sys/poll.h>
#include <sys/wait.h>
#include <sys/resource.h>
#include <sys/mman.h>
#include <sys/note.h>
#include <door.h>

#include <wchar.h>
#include <locale.h>
#include <stdarg.h>

#include "dataq.h"
#include "conf.h"
#include "syslogd.h"

#define DOORFILE                "/var/run/syslog_door"
#define RELATIVE_DOORFILE       "../var/run/syslog_door"
#define OLD_DOORFILE            "/etc/.syslog_door"

#define PIDFILE                 "/var/run/syslog.pid"
#define RELATIVE_PIDFILE        "../var/run/syslog.pid"
#define OLD_PIDFILE             "/etc/syslog.pid"

static char             *LogName = "/dev/log";
static char             *ConfFile = "/etc/syslog.conf";
static char             ctty[] = "/dev/console";
static char             sysmsg[] = "/dev/sysmsg";
static int              DoorFd = -1;
static int              DoorCreated = 0;
static int              PidfileCreated = 0;
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 nfds;
616         int timeout;
617         static int klogerrs = 0;
618         pthread_t mythreadno;
```



```
620         if (Debug) {
621                 mythreadno = pthread_self();
622         }

624         DPRINT1(1, "sys_poll(%u): sys_thread started\n", mythreadno);

626         /*
627          * Try to process as many messages as we can without blocking on poll.
628          * We identify such messages with timeout == 0 and
621          * We count such "initial" messages with sys_init_msg_count and
629          * enqueue them without the SYNC_FILE flag.  When no more data is
630          * waiting on the local log device, we set timeout to INFTIM,
631          * and generate a flush message to sync
632          * the previously identified messages out to disk.
624          * clear sys_init_msg_count, and generate a flush message to sync
625          * the previously counted initial messages out to disk.
633          */

635         timeout = INFTIM;
628         sys_init_msg_count = 0;

637         for (;;) {
638                 errno = 0;
639                 t_errno = 0;

641                 nfds = poll(&Pfd, 1, timeout);
634                 nfds = poll(&Pfd, 1, INFTIM);

643                 if (nfds <= 0) {
644                         if (nfds < 0 && errno != EINTR)
645                                 logerror("poll");
646                         if (timeout == 0)
647                                 flushmsg(SYNC_FILE);
648                         timeout = INFTIM;
636                 if (nfds == 0)
649                         continue;
650                 }

652                 if (Pfd.revents & POLLIN) {
653                         timeout = 0;
654                         getkmsg(timeout);
639                 if (nfds < 0) {
640                         if (errno != EINTR)
641                                 logerror("poll");
655                         continue;
656                 }

644                 if (Pfd.revents & POLLIN) {
645                         getkmsg(INFTIM);
646                 } else {
658                 if (shutting_down) {
659                         pthread_exit(0);
660                 }

662                 if (Pfd.revents & (POLLNVAL|POLLHUP|POLLERR)) {
663                         logerror("kernel log driver poll error");
664                         (void) close(Pfd.fd);
665                         Pfd.fd = -1;
666                 }
655                 }

668                 while (Pfd.fd == -1 && klogerrs++ < 10) {
669                         Pfd.fd = openklog(LogName, O_RDONLY);
670                 }
671                 if (klogerrs >= 10) {
```

```
672                         logerror("can't reopen kernel log device - fatal");
673                         exit(1);
674                 }
675                 if (timeout == 0)
676                         flushmsg(SYNC_FILE);
677                 timeout = INFTIM;
678         }
679         /*NOTREACHED*/
680         return (NULL);
681 }

683 /*
684  * Pull up one message from log driver.
685  */
686 static void
687 getkmsg(int timeout)
688 {
689         int flags = 0, i;
690         char *lastline;
691         struct strbuf ctl, dat;
692         struct log_ctl hdr;
693         char buf[MAXLINE+1];
694         size_t buflen;
695         size_t len;
696         char tmpbuf[MAXLINE+1];
697         pthread_t mythreadno;

699         if (Debug) {
700                 mythreadno = pthread_self();
701         }

703         dat.maxlen = MAXLINE;
704         dat.buf = buf;
705         ctl.maxlen = sizeof (struct log_ctl);
706         ctl.buf = (caddr_t)&hdr;

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

712                 DPRINT2(5, "getkmsg:(%u): getmsg: dat.len = %d\n",
698                 DPRINT2(5, "sys_poll:(%u): getmsg: dat.len = %d\n",
713                     mythreadno, dat.len);
714                 buflen = strlen(buf);
715                 len = findnl_bkwd(buf, buflen);

717                 (void) memcpy(tmpbuf, buf, len);
718                 tmpbuf[len] = '\0';

720                 /*
721                  * Format sys will enqueue the log message.
722                  * Set the sync flag if timeout != 0, which
723                  * means that we're done handling all the
724                  * initial messages ready during startup.
725                  */
726                 if (timeout == 0) {
727                         formatsys(&hdr, tmpbuf, 0);
728                         sys_init_msg_count++;
729                 } else {
730                         formatsys(&hdr, tmpbuf, 1);
731                 }
732                 sys_msg_count++;

734                 if (len != buflen) {
735                         /* If anything remains in buf */
736                         size_t remlen;
```



```
738                         if (buf[len] == '\n') {
739                                 /* skip newline */
740                                 len++;
741                         }

743                         /*
744                          *  Move the remaining bytes to
745                          * the beginnning of buf.
746                          */

748                         remlen = buflen - len;
749                         (void) memcpy(buf, &buf[len], remlen);
750                         dat.maxlen = MAXLINE - remlen;
751                         dat.buf = &buf[remlen];
752                 } else {
753                         dat.maxlen = MAXLINE;
754                         dat.buf = buf;
755                 }
756         }

758         if (i == 0 && dat.len > 0) {
759                 dat.buf[dat.len] = '\0';
760                 /*
761                  * Format sys will enqueue the log message.
762                  * Set the sync flag if timeout != 0, which
763                  * means that we're done handling all the
764                  * initial messages ready during startup.
765                  */
766                 DPRINT2(5, "getkmsg(%u): getmsg: dat.maxlen = %d\n",
767                     mythreadno, dat.maxlen);
768                 DPRINT2(5, "getkmsg(%u): getmsg: dat.len = %d\n",
769                     mythreadno, dat.len);
770                 DPRINT2(5, "getkmsg(%u): getmsg: strlen(dat.buf) = %d\n",
771                     mythreadno, strlen(dat.buf));
772                 DPRINT2(5, "getkmsg(%u): getmsg: dat.buf = \"%s\"\n",
773                     mythreadno, dat.buf);
774                 DPRINT2(5, "getkmsg(%u): buf len = %d\n",
775                     mythreadno, strlen(buf));
776                 if (timeout == 0) {
777                         formatsys(&hdr, buf, 0);
778                         sys_init_msg_count++;
779                 } else {
780                         formatsys(&hdr, buf, 1);
781                 }
782                 sys_msg_count++;
783         } else if (i < 0 && errno != EINTR) {
784                 if (!shutting_down) {
785                         logerror("kernel log driver read error");
786                 }
787                 (void) close(Pfd.fd);
788                 Pfd.fd = -1;
789         }
790 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   10215 Mon Oct 28 12:46:17 2013
new/usr/src/man/man1m/syslogd.1m
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
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"
   6 .TH SYSLOGD 1M "Oct 31, 2008"
   8 .SH NAME
   9 syslogd \- log system messages
  10 .SH SYNOPSIS
  11 .LP
  12 .nf
  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, \fBufs\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.
  65 See .
  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 \fB-log_from_remote\fR is \fBfalse\fR. See NOTES, below.
  89 .sp
  90 .ne 2
  91 .na
  92 \fB\fBLOG_FROM_REMOTE\fR\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\fB-d\fR\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\fB-f\fR \fIconfigfile\fR\fR
 122 .ad
 123 .sp .6
```

```
.RS 4n
Specify an alternate configuration file.
.RE

.sp
.ne 2
.na
\fB\fB-m\fR \fImarkinterval\fR\fR
.ad
.sp .6
.RS 4n
Specify an interval, in minutes, between mark messages.
.RE

.sp
.ne 2
.na
\fB\fB-p\fR \fIpath\fR\fR
.ad
.sp .6
.RS 4n
Specify an alternative log device name. The default is \fB/dev/log\fR.
.RE

.sp
.ne 2
.na
\fB\fB-T\fR\fR
.ad
.sp .6
.RS 4n
Enable the \fBsyslogd\fR \fBUDP\fR port to turn on logging of remote messages.
This is the default behavior. See \fBEXAMPLES\fR.
This is the default behavior. See .
.RE

.sp
.ne 2
.na
\fB\fB-t\fR\fR
.ad
.sp .6
.RS 4n
Disable the \fBsyslogd\fR \fBUDP\fR port to turn off logging of remote
messages. See \fBEXAMPLES\fR.
messages. See .
.RE

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

.sp
.in +2
.nf
Sep 29 21:41:18 cathy unix: alloc /: file system full
.fi
.in -2
.sp

.LP
\fBExample 2 \fR\fBsyslogd\fR Output with ID generation Enabled
```



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

.sp
.in +2
.nf
Sep 29 21:41:18 cathy ufs: [ID 845546 kern.notice]
                                        alloc /: file system full
.fi
.in -2
.sp

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

.sp
.in +2
.nf
Sep 29 21:41:18 cathy ufs: alloc /: file system full
.fi
.in -2
.sp

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

.sp
.in +2
.nf
# \fBsvccfg -s svc:/system/system-log setprop config/log_from_remote = true\fR
# \fBsvcadm restart svc:/system/system-log\fR
.fi
.in -2
.sp

.SH FILES
.sp
.ne 2
.na
\fB\fB/etc/syslog.conf\fR\fR
.ad
.sp .6
.RS 4n
Configuration file
.RE

.sp
.ne 2
.na
\fB\fB/var/run/syslog.pid\fR\fR
.ad
.sp .6
.RS 4n
Process \fBID\fR
```

```
.RE

.sp
.ne 2
.na
\fB\fB/etc/default/syslogd\fR\fR
.ad
.sp .6
.RS 4n
Contains default settings. You can override some of the settings by
command-line options.
.RE

.sp
.ne 2
.na
\fB\fB/dev/log\fR\fR
.ad
.sp .6
.RS 4n
\fBSTREAMS\fR log driver
.RE

.sp
.ne 2
.na
\fB\fB/etc/netconfig\fR\fR
.ad
.sp .6
.RS 4n
Transport providers available on the system
.RE

.sp
.ne 2
.na
\fB\fB/etc/net/transport/hosts\fR\fR
.ad
.sp .6
.RS 4n
Network hosts for each transport
.RE

.sp
.ne 2
.na
\fB\fB/etc/net/transport/services\fR\fR
.ad
.sp .6
.RS 4n
Network services for each transport
.RE

.SH SEE ALSO
.sp
.LP
\fBlogger\fR(1), \fBsvcs\fR(1), \fBmsgid\fR(1M), \fBsvcadm\fR(1M),
\fBsvccfg\fR(1M), \fBsyslog\fR(3C), \fBsyslog.conf\fR(4), \fBattributes\fR(5),
\fBsignal.h\fR(3HEAD), \fBsmf\fR(5), \fBlog\fR(7D)
.SH NOTES
.sp
.LP
The \fBmark\fR message is a system time stamp, and so it is only defined for
the system on which \fBsyslogd\fR is running. It can not be forwarded to other
systems.
.sp
```



```
.LP
When \fBsyslogd\fR receives a \fBHUP\fR signal, it attempts to complete
outputting pending messages, and close all log files to which it is currently
logging messages. If, for some reason, one (or more) of these files does not
close within a generous grace period, \fBsyslogd\fR discards the pending
messages, forcibly closes these files, and starts reconfiguration. If this
shutdown procedure is disturbed by an unexpected error and \fBsyslogd\fR cannot
complete reconfiguration, \fBsyslogd\fR sends a mail message to the superuser
on the current system stating that it has shut down, and exits.
.sp
.LP
Care should be taken to ensure that each window displaying messages forwarded
by \fBsyslogd\fR (especially console windows) is run in the system default
locale (which is \fBsyslogd\fR's locale). If this advice is not followed, it is
possible for a \fBsyslog\fR message to alter the terminal settings for that
window, possibly even allowing remote execution of arbitrary commands from that
window.
.sp
.LP
The \fBsyslogd\fR service is managed by the service management facility,
\fBsmf\fR(5), under the service identifier:
.sp
.in +2
.nf
 svc:/system/system-log:default
.fi
.in -2
.sp

.sp
.LP
Administrative actions on this service, such as enabling, disabling, or
requesting restart, can be performed using \fBsvcadm\fR(1M). The service's
status can be queried using the \fBsvcs\fR(1) command.
.sp
.LP
When \fBsyslogd\fR is started by means of \fBsvcadm\fR(1M), if a value is
specified for \fBLOG_FROM_REMOTE\fR in the \fB/etc/defaults/syslogd\fR file,
the SMF property \fBsvc:/system/system-log/config/log_from_remote\fR is set to
correspond to the \fBLOG_FROM_REMOTE\fR value and the
\fB/etc/default/syslogd\fR file is modified to replace the
\fBLOG_FROM_REMOTE\fR specification with the following comment:
.sp
.in +2
.nf
# LOG_FROM_REMOTE is now set using svccfg(1m), see syslogd(1m).
.fi
.in -2

.sp
.LP
If neither \fBLOG_FROM_REMOTE\fR nor
\fBsvc:/system/system-log/config/log_from_remote\fR are defined, the default is
to log remote messages.
.sp
.LP
On installation, the initial value of
\fBsvc:/system/system-log/config/log_from_remote\fR is \fBfalse\fR.
```

```
**********************************************************
    8168 Mon Oct 28 12:46:17 2013
new/usr/src/man/man4/syslog.conf.4
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
1762 Syslogd man page: missing reference.
**********************************************************
'\" te
.\" Copyright (c) 2013 Gary Mills
.\" Copyright (c) 2003 Sun Microsystems, Inc. All Rights Reserved.
.\" Copyright (c) 1983 Regents of the University of California.  All rights rese
.TH SYSLOG.CONF 4 "Oct 17, 2013"
.TH SYSLOG.CONF 4 "Apr 26, 2006"
.SH NAME
syslog.conf \- configuration file for syslogd system log daemon
.SH SYNOPSIS
.LP
.nf
\fB/etc/syslog.conf\fR
.fi

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

.sp
.LP
The \fIselector\fR field contains a semicolon-separated list of priority
specifications of the form:
.sp
.in +2
.nf
\fIfacility\fR\fB\&.\fR\fIlevel\fR [ \fB;\fR \fIfacility\fR\fB\&.\fR\fIlevel\fR
.fi
.in -2

.sp
.LP
where \fIfacility\fR is a system facility, or comma-separated list of
facilities, and \fIlevel\fR is an indication of the severity of the condition
being logged. Recognized values for \fIfacility\fR include:
.sp
.ne 2
.na
\fB\fBkern\fR\fR
\fIfB\fBuser\fR\fR
.ad
.RS 12n
Messages generated by the kernel.
Messages generated by user processes. This is the default priority for messages
from programs or facilities not listed in this file.
```



```
.RE

.sp
.ne 2
.na
\fB\fBuser\fR\fR
\fIfB\fBkern\fR\fR
.ad
.RS 12n
Messages generated by user processes. This is the default priority for messages
from programs or facilities not listed in this file.
Messages generated by the kernel.
.RE

.sp
.ne 2
.na
\fB\fBmail\fR\fR
.ad
.RS 12n
The mail system.
.RE

.sp
.ne 2
.na
\fB\fBdaemon\fR\fR
.ad
.RS 12n
System daemons, such as \fBin.ftpd\fR(1M)
.RE

.sp
.ne 2
.na
\fB\fBauth\fR\fR
.ad
.RS 12n
The authorization system: \fBlogin\fR(1), \fBsu\fR(1M), \fBgetty\fR(1M), among
others.
.RE

.sp
.ne 2
.na
\fB\fBlpr\fR\fR
.ad
.RS 12n
The line printer spooling system: \fBlpr\fR(1B), \fBlpc\fR(1B), among others.
.RE

.sp
.ne 2
.na
\fB\fBnews\fR\fR
.ad
.RS 12n
Designated for the USENET network news system.
.RE

.sp
.ne 2
.na
\fB\fBuucp\fR\fR
.ad
.RS 12n
```

```
Designated for the UUCP system; it does not currently use the \fBsyslog\fR
mechanism.
.RE

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

.sp
.ne 2
.na
\fB\fBauthpriv\fR\fR
.ad
.RS 12n
Designated for the BSD security/authorization system.
.RE

.sp
.ne 2
.na
\fB\fBftp\fR\fR
.ad
.RS 12n
Designated for the file transfer system.
.RE

.sp
.ne 2
.na
\fB\fBntp\fR\fR
.ad
.RS 12n
Designated for the network time system.
.RE

.sp
.ne 2
.na
\fB\fBaudit\fR\fR
.ad
.RS 12n
Designated for audit messages generated by systems that audit by means of
syslog.
.RE

.sp
.ne 2
.na
\fB\fBconsole\fR\fR
.ad
.RS 12n
Designated for the BSD console system.
.RE

.sp
.ne 2
.na
```



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

.sp
.ne 2
.na
\fB\fBlocal0-7\fR\fR
.ad
.RS 12n
Designated for local use.
.RE

.sp
.ne 2
.na
\fB\fBmark\fR\fR
.ad
.RS 12n
For timestamp messages produced internally by \fBsyslogd\fR.
.RE

.sp
.ne 2
.na
\fB\fB*\fR\fR
.ad
.RS 12n
An asterisk indicates all facilities except for the \fBmark\fR facility.
.RE

.sp
.LP
Recognized values for \fIlevel\fR are (in descending order of severity):
.sp
.ne 2
.na
\fB\fBemerg\fR\fR
.ad
.RS 11n
For panic conditions that would normally be broadcast to all users.
.RE

.sp
.ne 2
.na
\fB\fBalert\fR\fR
.ad
.RS 11n
For conditions that should be corrected immediately, such as a corrupted system
database.
.RE

.sp
.ne 2
.na
\fB\fBcrit\fR\fR
.ad
.RS 11n
For warnings about critical conditions, such as hard device errors.
.RE
```

```
.sp
.ne 2
.na
\fB\fBerr\fR\fR
.ad
.RS 11n
For other errors.
.RE

.sp
.ne 2
.na
\fB\fBwarning\fR\fR
.ad
.RS 11n
For warning messages.
.RE

.sp
.ne 2
.na
\fB\fBnotice\fR\fR
.ad
.RS 11n
For conditions that are not error conditions, but may require special handling.
A configuration entry with a \fIlevel\fR value of \fBnotice\fR must appear on a
separate line.
.RE

.sp
.ne 2
.na
\fB\fBinfo\fR\fR
.ad
.RS 11n
Informational messages.
.RE

.sp
.ne 2
.na
\fB\fBdebug\fR\fR
.ad
.RS 11n
For messages that are normally used only when debugging a program.
.RE

.sp
.ne 2
.na
\fB\fBnone\fR\fR
.ad
.RS 11n
Do not send messages from the indicated \fIfacility\fR to the selected file.
For example, a \fIselector\fR of
.sp
\fB*.debug;mail.none\fR
.sp
sends all messages \fIexcept\fR mail messages to the selected file.
.RE

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



```
levels.
.sp
.LP
The \fIaction\fR field indicates where to forward the message. Values for this
field can have one of four forms:
.RS +4
.TP
.ie t \(bu
.el o
A filename, beginning with a leading slash, which indicates that messages
specified by the \fIselector\fR are to be written to the specified file. The
file is opened in append mode if it exists. If the file does not exist, logging
silently fails for this action.
.RE
.RS +4
.TP
.ie t \(bu
.el o
The name of a remote host, prefixed with an \fB@\fR, as with:
\fB@\fR\fIserver\fR, which indicates that messages specified by the
\fIselector\fR are to be forwarded to the \fBsyslogd\fR on the named host. The
hostname "loghost" is treated, in the default \fBsyslog.conf\fR, as the
hostname given to the machine that logs \fBsyslogd\fR messages. Every machine
is "loghost" by default, per the hosts database. It is also possible to specify
one machine on a network to be "loghost" by, literally, naming the machine
"loghost". If the local machine is designated to be "loghost", then
\fBsyslogd\fR messages are written to the appropriate files. Otherwise, they
are sent to the machine "loghost" on the network.
.RE
.RS +4
.TP
.ie t \(bu
.el o
A comma-separated list of usernames, which indicates that messages specified by
the \fIselector\fR are to be written to the named users if they are logged in.
.RE
.RS +4
.TP
.ie t \(bu
.el o
An asterisk, which indicates that messages specified by the \fIselector\fR are
to be written to all logged-in users.
.RE
.sp
.LP
Blank lines are ignored. Lines for which the first nonwhite character is
a '\fB#\fR' are treated as comments.
.SH EXAMPLES
.LP
\fBExample 1 \fRA Sample Configuration File
.sp
.LP
With the following configuration file:

.sp

.sp
.TS
l l
l l .
\fB*.notice\fR  \fB/var/log/notice\fR
\fBmail.info\fR \fB/var/log/notice\fR
\fB*.crit\fR    \fB/var/log/critical\fR
\fBkern,mark.debug\fR   \fB/dev/console\fR
\fBkern.err\fR  \fB@server\fR
\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 Oct 28 12:46:17 2013
new/usr/src/uts/common/os/logsubr.c
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
1762 Syslogd man page: missing reference.
***********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

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

#include <sys/types.h>
#include <sys/param.h>
#include <sys/varargs.h>
#include <sys/systm.h>
#include <sys/cmn_err.h>
#include <sys/stream.h>
#include <sys/strsubr.h>
#include <sys/strsun.h>
#include <sys/sysmacros.h>
#include <sys/kmem.h>
#include <sys/log.h>
#include <sys/spl.h>
#include <sys/syslog.h>
#include <sys/console.h>
#include <sys/debug.h>
#include <sys/utsname.h>
#include <sys/id_space.h>
#include <sys/zone.h>

log_zone_t log_global;
queue_t *log_consq;
queue_t *log_backlogq;
queue_t *log_intrq;

#define LOG_PRISIZE     8       /* max priority size: 7 characters + null */
#define LOG_FACSIZE     9       /* max priority size: 8 characters + null */

static krwlock_t log_rwlock;
static int log_rwlock_depth;
static int log_seq_no[SL_CONSOLE + 1];
static stdata_t log_fakestr;
static id_space_t *log_minorspace;
static log_t log_backlog;
```



```
static struct kmem_cache *log_cons_cache;       /* log_t cache */

static queue_t *log_recentq;
static queue_t *log_freeq;

static zone_key_t log_zone_key;

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

static char log_pri[LOG_PRIMASK + 1][LOG_PRISIZE] = {
        "emerg",        "alert",        "crit",         "error",
        "warning",      "notice",       "info",         "debug"
};

static char log_fac[LOG_NFACILITIES + 1][LOG_FACSIZE] = {
        "kern",         "user",         "mail",         "daemon",
        "auth",         "syslog",       "lpr",          "news",
        "uucp",         "bsdcron",      "authpriv",     "ftp",
        "ntp",          "audit",        "console",      "cron",
        "uucp",         "resv9",        "resv10",       "resv11",
        "resv12",       "audit",        "resv14",       "cron",
        "local0",       "local1",       "local2",       "local3",
        "local4",       "local5",       "local6",       "local7",
        "unknown"
};
```

_____*unchanged_portion_omitted_*

```
**********************************************************
    3229 Mon Oct 28 12:46:17 2013
new/usr/src/uts/common/sys/syslog.h
4211 Some syslog facility names and symbols are missing
3232 syslogd shouldn't sync after each LOG_KERN line
1762 Syslogd man page: missing reference.
**********************************************************
```

```
/*
 * Copyright (c) 2013 Gary Mills
 * Copyright 2003 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

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

/*
 * University Copyright- Copyright (c) 1982, 1986, 1988
 * The Regents of the University of California
 * All Rights Reserved
 *
 * University Acknowledgment- Portions of this document are derived from
 * software developed by the University of California, Berkeley, and its
 * contributors.
 */

#ifndef _SYS_SYSLOG_H
#define _SYS_SYSLOG_H

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

#ifdef  __cplusplus
extern "C" {
#endif

/*
 *  Facility codes
 */
#define LOG_KERN        (0<<3)  /* kernel messages */
#define LOG_USER        (1<<3)  /* random user-level messages */
#define LOG_MAIL        (2<<3)  /* mail system */
#define LOG_DAEMON      (3<<3)  /* system daemons */
#define LOG_AUTH        (4<<3)  /* security/authorization messages */
#define LOG_SYSLOG      (5<<3)  /* messages generated internally by syslogd */
#define LOG_LPR         (6<<3)  /* line printer subsystem */
#define LOG_NEWS        (7<<3)  /* netnews subsystem */
#define LOG_UUCP        (8<<3)  /* uucp subsystem */
#define LOG_BSDCRON     (9<<3)  /* BSD cron/at subsystem */
#define LOG_AUTHPRIV    (10<<3) /* BSD security/authorization messages */
#define LOG_FTP         (11<<3) /* file transfer subsystem */
#define LOG_NTP         (12<<3) /* network time subsystem */
#define LOG_AUDIT       (13<<3) /* audit subsystem */
#define LOG_CONSOLE     (14<<3) /* BSD console messages */
#define LOG_CRON        (15<<3) /* cron/at subsystem */
        /* other codes through 15 reserved for system use */
#define LOG_LOCAL0      (16<<3) /* reserved for local use */
#define LOG_LOCAL1      (17<<3) /* reserved for local use */
#define LOG_LOCAL2      (18<<3) /* reserved for local use */
#define LOG_LOCAL3      (19<<3) /* reserved for local use */
#define LOG_LOCAL4      (20<<3) /* reserved for local use */
#define LOG_LOCAL5      (21<<3) /* reserved for local use */
#define LOG_LOCAL6      (22<<3) /* reserved for local use */
#define LOG_LOCAL7      (23<<3) /* reserved for local use */

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



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

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

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

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

#ifdef  __cplusplus
}
_____unchanged_portion_omitted_
```