

```

*****
115298 Sat Jun 1 10:02:47 2013
new/usr/src/cmd/init/init.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
25 * Copyright (c) 1988, 2010, Oracle and/or its affiliates. All rights reserved.
26 */

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

31 /*
32 * University Copyright- Copyright (c) 1982, 1986, 1988
33 * The Regents of the University of California
34 * All Rights Reserved
35 *
36 * University Acknowledgment- Portions of this document are derived from
37 * software developed by the University of California, Berkeley, and its
38 * contributors.
39 */

41 /*
42 * init(1M) is the general process spawning program. Its primary job is to
43 * start and restart svc.startd for smf(5). For backwards-compatibility it also
44 * spawns and respawns processes according to /etc/inittab and the current
45 * run-level. It reads /etc/default/inittab for general configuration.
46 *
47 * To change run-levels the system administrator runs init from the command
48 * line with a level name. init signals svc.startd via libscf and directs the
49 * zone's init (pid 1 in the global zone) what to do by sending it a signal;
50 * these signal numbers are commonly referred to in the code as 'states'. Valid
51 * run-levels are [sS0123456]. Additionally, init can be given directives
52 * [qQabc], which indicate actions to be taken pertaining to /etc/inittab.
53 *
54 * When init processes inittab entries, it finds processes that are to be
55 * spawned at various run-levels. inittab contains the set of the levels for
56 * which each inittab entry is valid.
57 *
58 * State File and Restartability
59 * Premature exit by init(1M) is handled as a special case by the kernel:
60 * init(1M) will be immediately re-executed, retaining its original PID. (PID

```

```

61 * 1 in the global zone.) To track the processes it has previously spawned,
62 * as well as other mutable state, init(1M) regularly updates a state file
63 * such that its subsequent invocations have knowledge of its various
64 * dependent processes and duties.
65 *
66 * Process Contracts
67 * We start svc.startd(1M) in a contract and transfer inherited contracts when
68 * restarting it. Everything else is started using the legacy contract
69 * template, and the created contracts are abandoned when they become empty.
70 *
71 * utmpx Entry Handling
72 * Because init(1M) no longer governs the startup process, its knowledge of
73 * when utmpx becomes writable is indirect. However, spawned processes
74 * expect to be constructed with valid utmpx entries. As a result, attempts
75 * to write normal entries will be retried until successful.
76 *
77 * Maintenance Mode
78 * In certain failure scenarios, init(1M) will enter a maintenance mode, in
79 * which it invokes sulogin(1M) to allow the operator an opportunity to
80 * repair the system. Normally, this operation is performed as a
81 * fork(2)-exec(2)-waitpid(3C) sequence with the parent waiting for repair or
82 * diagnosis to be completed. In the cases that fork(2) requests themselves
83 * fail, init(1M) will directly execute sulogin(1M), and allow the kernel to
84 * restart init(1M) on exit from the operator session.
85 *
86 * One scenario where init(1M) enters its maintenance mode is when
87 * svc.startd(1M) begins to fail rapidly, defined as when the average time
88 * between recent failures drops below a given threshold.
89 */

91 #include <sys/contract/process.h>
92 #include <sys/ctfs.h>
93 #include <sys/stat.h>
94 #include <sys/statvfs.h>
95 #include <sys/stropts.h>
96 #include <sys/systeminfo.h>
97 #include <sys/time.h>
98 #include <sys/termios.h>
99 #include <sys/tty.h>
100 #include <sys/types.h>
101 #include <sys/utsname.h>

103 #include <bsm/adt_event.h>
104 #include <bsm/libbsm.h>
105 #include <security/pam_appl.h>

107 #include <assert.h>
108 #include <ctype.h>
109 #include <dirent.h>
110 #include <errno.h>
111 #include <fcntl.h>
112 #include <libcontract.h>
113 #include <libcontract_priv.h>
114 #include <libintl.h>
115 #include <libscf.h>
116 #include <libscf_priv.h>
117 #include <poll.h>
118 #include <procfs.h>
119 #include <signal.h>
120 #include <stdarg.h>
121 #include <stdio.h>
122 #include <stdio_ext.h>
123 #include <stdlib.h>
124 #include <string.h>
125 #include <strings.h>
126 #include <syslog.h>

```

```

127 #include <time.h>
128 #include <ulimit.h>
129 #include <unistd.h>
130 #include <utmpx.h>
131 #include <wait.h>
132 #include <zone.h>
133 #include <ucontext.h>

135 #undef sleep

137 #define fiocctl(p, sptr, cmd)   ioctl(fileno(p), sptr, cmd)
138 #define min(a, b)              ((a) < (b)) ? (a) : (b)

140 #define TRUE    1
141 #define FALSE  0
142 #define FAILURE -1

144 #define UT_USER_SZ    32    /* Size of a utmpx ut_user field */
145 #define UT_LINE_SZ   32    /* Size of a utmpx ut_line field */

147 /*
148  * SLEEPTIME    The number of seconds "init" sleeps between wakeups if
149  *              nothing else requires this "init" wakeup.
150  */
151 #define SLEEPTIME    (5 * 60)

153 /*
154  * MAXCMDL     The maximum length of a command string in inittab.
155  */
156 #define MAXCMDL     512

158 /*
159  * EXEC        The length of the prefix string added to all comamnds
160  *              found in inittab.
161  */
162 #define EXEC        (sizeof ("exec ") - 1)

164 /*
165  * TWARN       The amount of time between warning signal, SIGTERM,
166  *              and the fatal kill signal, SIGKILL.
167  */
168 #define TWARN       5

170 #define id_eq(x, y)    ((x[0] == y[0] && x[1] == y[1] && x[2] == y[2] &&
171                       x[3] == y[3]) ? TRUE : FALSE)

173 /*
174  * The kernel's default umask is 022 these days; since some processes inherit
175  * their umask from init, init will set it from CMASK in /etc/default/init.
176  * init gets the default umask from the kernel, it sets it to 022 whenever
177  * it wants to create a file and reverts to CMASK afterwards.
178  */

180 static int cmask;

182 /*
183  * The following definitions, concluding with the 'lvls' array, provide a
184  * common mapping between level-name (like 'S'), signal number (state),
185  * run-level mask, and specific properties associated with a run-level.
186  * This array should be accessed using the routines lvlname_to_state(),
187  * lvlname_to_mask(), state_to_mask(), and state_to_flags().
188  */

190 /*
191  * Correspondence of signals to init actions.
192  */

```

```

193 #define LVLQ          SIGHUP
194 #define LVL0          SIGINT
195 #define LVL1          SIGQUIT
196 #define LVL2          SIGILL
197 #define LVL3          SIGTRAP
198 #define LVL4          SIGIOT
199 #define LVL5          SIGEMT
200 #define LVL6          SIGFPE
201 #define SINGLE_USER  SIGBUS
202 #define LVLa          SIGSEGV
203 #define LVLb          SIGSYS
204 #define LVLc          SIGPIPE

206 /*
207  * Bit Mask for each level.  Used to determine legal levels.
208  */
209 #define MASK0    0x0001
210 #define MASK1    0x0002
211 #define MASK2    0x0004
212 #define MASK3    0x0008
213 #define MASK4    0x0010
214 #define MASK5    0x0020
215 #define MASK6    0x0040
216 #define MASKSU   0x0080
217 #define MASKa    0x0100
218 #define MASKb    0x0200
219 #define MASKc    0x0400

221 #define MASK_NUMERIC (MASK0 | MASK1 | MASK2 | MASK3 | MASK4 | MASK5 | MASK6)
222 #define MASK_abc (MASKa | MASKb | MASKc)

224 /*
225  * Flags to indicate properties of various states.
226  */
227 #define LSEL_RUNLEVEL    0x0001 /* runlevels you can transition to */

229 typedef struct lvl {
230     int    lvl_state;
231     int    lvl_mask;
232     char   lvl_name;
233     int    lvl_flags;
234 } lvl_t;
unchanged portion omitted

1506 /*
1507  * getcmd() parses lines from inittab.  Each time it finds a command line
1508  * it will return TRUE as well as fill the passed CMD_LINE structure and
1509  * the shell command string.  When the end of inittab is reached, FALSE
1510  * is returned inittab is automatically opened if it is not currently open
1511  * and is closed when the end of the file is reached.
1512  */
1513 static FILE *fp_inittab = NULL;

1515 static int
1516 getcmd(struct CMD_LINE *cmd, char *shcmd)
1517 {
1518     char   *ptr;
1519     int    c, lastc, state;
1520     char   *ptr1;
1521     int    answer, i, proceed;
1522     struct stat sbuff;
1523     static char *actions[] = {
1524         "off", "respawn", "ondemand", "once", "wait", "boot",
1525         "bootwait", "powerfail", "powerwait", "initdefault",
1526         "sysinit",
1527     };
};

```

```

1528 static short act_masks[] = {
1529     M_OFF, M_RESPAWN, M_ONDEMAND, M_ONCE, M_WAIT, M_BOOT,
1530     M_BOOTWAIT, M_PF, M_PWAIT, M_INITDEFAULT, M_SYSINIT,
1531 };
1532 /*
1533  * Only these actions will be allowed for entries which
1534  * are specified for single-user mode.
1535  */
1536 short su_acts = M_INITDEFAULT | M_PF | M_PWAIT | M_WAIT;

1538 if (fp_inittab == NULL) {
1539     /*
1540      * Before attempting to open inittab we stat it to make
1541      * sure it currently exists and is not empty. We try
1542      * several times because someone may have temporarily
1543      * unlinked or truncated the file.
1544      */
1545     for (i = 0; i < 3; i++) {
1546         if (stat(INITTAB, &sbuf) == -1) {
1547             if (i == 2) {
1548                 console(B_TRUE,
1549                     "Cannot stat %s, errno: %d\n",
1550                     INITTAB, errno);
1551                 return (FAILURE);
1552             } else {
1553                 timer(3);
1554             }
1555         } else if (sbuf.st_size < 10) {
1556             if (i == 2) {
1557                 console(B_TRUE,
1558                     "%s truncated or corrupted\n",
1559                     INITTAB);
1560                 return (FAILURE);
1561             } else {
1562                 timer(3);
1563             }
1564         } else {
1565             break;
1566         }
1567     }

1569     /*
1570      * If unable to open inittab, print error message and
1571      * return FAILURE to caller.
1572      */
1573     if ((fp_inittab = fopen(INITTAB, "r")) == NULL) {
1574         console(B_TRUE, "Cannot open %s errno: %d\n", INITTAB,
1575             errno);
1576         return (FAILURE);
1577     }
1578 }

1580 /*
1581  * Keep getting commands from inittab until you find a
1582  * good one or run out of file.
1583  */
1584 for (answer = FALSE; answer == FALSE; ) {
1585     /*
1586      * Zero out the cmd itself before trying next line.
1587      */
1588     bzero(cmd, sizeof (struct CMD_LINE));

1590     /*
1591      * Read in lines of inittab, parsing at colons, until a line is
1592      * read in which doesn't end with a backslash. Do not start if
1593      * the first character read is an EOF. Note that this means

```

```

1594     * that lines which don't end in a newline are still processed,
1595     * since the "for" will terminate normally once started,
1596     * regardless of whether line terminates with a newline or EOF.
1597     */
1598     state = FAILURE;
1599     if ((c = fgetc(fp_inittab)) == EOF) {
1600         answer = FALSE;
1601         (void) fclose(fp_inittab);
1602         fp_inittab = NULL;
1603         break;
1604     }

1606     for (proceed = TRUE, ptr = shcmd, state = ID, lastc = '\0';
1607         proceed && c != EOF;
1608         lastc = c, c = fgetc(fp_inittab)) {
1609         /* If we're not in the FAILURE state and haven't */
1610         /* yet reached the shell command field, process */
1611         /* the line, otherwise just look for a real end */
1612         /* of line. */
1613         if (state != FAILURE && state != COMMAND) {
1614             /*
1615              * Squeeze out spaces and tabs.
1616              */
1617             if (c == ' ' || c == '\t')
1618                 continue;

1620             /*
1621              * Ignore characters in a comment, except for the \n.
1622              */
1623             if (state == COMMENT) {
1624                 if (c == '\n') {
1625                     lastc = ' ';
1626                     break;
1627                 } else {
1628                     continue;
1629                 }
1630             }

1632             /*
1633              * Detect comments (lines whose first non-whitespace
1634              * character is '#') by checking that we're at the
1635              * beginning of a line, have seen a '#', and haven't
1636              * yet accumulated any characters.
1637              */
1638             if (state == ID && c == '#' && ptr == shcmd) {
1639                 state = COMMENT;
1640                 continue;
1641             }

1643             /*
1644              * If the character is a ':', then check the
1645              * previous field for correctness and advance
1646              * to the next field.
1647              */
1648             if (c == ':') {
1649                 switch (state) {

1651                     case ID :
1652                         /*
1653                          * Check to see that there are only
1654                          * 1 to 4 characters for the id.
1655                          */
1656                         if ((i = ptr - shcmd) < 1 || i > 4) {
1657                             state = FAILURE;
1658                         } else {
1659                             bcopy(shcmd, &cmd->c_id[0], i);

```

```

1660         ptr = shcmd;
1661         state = LEVELS;
1662     }
1663     break;

1665     case LEVELS :
1666     /*
1667      * Build a mask for all the levels for
1668      * which this command will be legal.
1669      */
1670     for (cmd->c_levels = 0, ptr1 = shcmd;
1671          ptr1 < ptr; ptr1++) {
1672         int mask;
1673         if (lvlname_to_mask(*ptr1,
1674                             &mask) == -1) {
1675             state = FAILURE;
1676             break;
1677         }
1678         cmd->c_levels |= mask;
1679     }
1680     if (state != FAILURE) {
1681         state = ACTION;
1682         ptr = shcmd; /* Reset the buffer */
1683     }
1684     break;

1686     case ACTION :
1687     /*
1688      * Null terminate the string in shcmd buffer and
1689      * then try to match against legal actions. If
1690      * the field is of length 0, then the default of
1691      * "RESPAWN" is used if the id is numeric,
1692      * otherwise the default is "OFF".
1693      */
1694     if (ptr == shcmd) {
1695         if (isdigit(cmd->c_id[0]) &&
1696             (cmd->c_id[1] == '\0' ||
1697              isdigit(cmd->c_id[1]) &&
1698              (cmd->c_id[2] == '\0' ||
1699               isdigit(cmd->c_id[2]) &&
1700                (cmd->c_id[3] == '\0' ||
1701                 isdigit(cmd->c_id[3]))))
1702             cmd->c_action = M_RESPAWN;
1703         else
1704             cmd->c_action = M_OFF;
1705     } else {
1706         for (cmd->c_action = 0, i = 0,
1707              *ptr = '\0';
1708              i <
1709              sizeof (actions)/sizeof (char *);
1710              for (cmd->c_action = 0, i = 0, *ptr = '\0';
1711                  i < sizeof (actions)/sizeof (char *);
1712                  i++) {
1713             if (strcmp(shcmd, actions[i]) == 0) {
1714                 if ((cmd->c_levels & MASKSU) &&
1715                     !(act_masks[i] & su_acts))
1716                     cmd->c_action = 0;
1717                 else
1718                     cmd->c_action =
1719                         act_masks[i];
1720                 cmd->c_action = act_masks[i];
1721                 break;
1722             }
1723         }
1724     }

```

```

1723     /*
1724      * If the action didn't match any legal action,
1725      * set state to FAILURE.
1726      */
1727     if (cmd->c_action == 0) {
1728         state = FAILURE;
1729     } else {
1730         state = COMMAND;
1731         (void) strcpy(shcmd, "exec ");
1732     }
1733     ptr = shcmd + EXEC;
1734     break;
1735 }
1736 }
1737 }
1738 }

1740 /* If the character is a '\n', then this is the end of a */
1741 /* line. If the '\n' wasn't preceded by a backslash, */
1742 /* it is also the end of an inittab command. If it was */
1743 /* preceded by a backslash then the next line is a */
1744 /* continuation. Note that the continuation '\n' falls */
1745 /* through and is treated like other characters and is */
1746 /* stored in the shell command line. */
1747 if (c == '\n' && lastc != '\\') {
1748     proceed = FALSE;
1749     *ptr = '\0';
1750     break;
1751 }

1753 /* For all other characters just stuff them into the */
1754 /* command as long as there aren't too many of them. */
1755 /* Make sure there is room for a terminating '\0' also. */
1756 if (ptr >= shcmd + MAXCMDL - 1)
1757     state = FAILURE;
1758 else
1759     *ptr++ = (char)c;

1761 /* If the character we just stored was a quoted */
1762 /* backslash, then change "c" to '\0', so that this */
1763 /* backslash will not cause a subsequent '\n' to appear */
1764 /* quoted. In otherwords '\ ' '\n' '\n' is the real end */
1765 /* of a command, while '\ ' '\n' is a continuation. */
1766 if (c == '\\ ' && lastc == '\\ ')
1767     c = '\0';
1768 }

1770 /*
1771  * Make sure all the fields are properly specified
1772  * for a good command line.
1773  */
1774 if (state == COMMAND) {
1775     answer = TRUE;
1776     cmd->c_command = shcmd;

1778     /*
1779      * If no default level was supplied, insert
1780      * all numerical levels.
1781      */
1782     if (cmd->c_levels == 0)
1783         cmd->c_levels = MASK_NUMERIC;

1785     /*
1786      * If no action has been supplied, declare this
1787      * entry to be OFF.
1788      */

```

```

1789         if (cmd->c_action == 0)
1790             cmd->c_action = M_OFF;

1792         /*
1793          * If no shell command has been supplied, make sure
1794          * there is a null string in the command field.
1795          */
1796         if (ptr == shcmd + EXEC)
1797             *shcmd = '\0';
1798     } else
1799         answer = FALSE;

1801     /*
1802     * If we have reached the end of inittab, then close it
1803     * and quit trying to find a good command line.
1804     */
1805     if (c == EOF) {
1806         (void) fclose(fp_inittab);
1807         fp_inittab = NULL;
1808         break;
1809     }
1810     return (answer);
1811 }
1812 }

```

unchanged portion omitted

```

2041 /*
2042 * boot_init(): Do initialization things that should be done at boot.
2043 */
2044 void
2045 boot_init()
2046 {
2047     int i;
2048     struct PROC_TABLE *process, *oprocess;
2049     struct CMD_LINE cmd;
2050     char line[MAXCMDL];
2051     char svc_aux[SVC_AUX_SIZE];
2052     char init_svc_fmri[SVC_FMRI_SIZE];
2053     char *old_path;
2054     int maxfiles;

2056     /* Use INIT_PATH for sysinit cmds */
2057     old_path = glob_envp[0];
2058     glob_envp[0] = malloc((unsigned)(strlen(INIT_PATH)+2));
2059     (void) strcpy(glob_envp[0], INIT_PATH);

2061     /*
2062     * Scan inittab(4) and process the special svc.startd entry, initdefault
2063     * and sysinit entries.
2064     */
2065     while (getcmd(&cmd, &line[0]) == TRUE) {
2066         if (startd_tmpl >= 0 && id_eq(cmd.c_id, "smf")) {
2067             process_startd_line(&cmd, line);
2068             (void) snprintf(startd_svc_aux, SVC_AUX_SIZE,
2069                 INITTAB_ENTRY_ID_STR_FORMAT, cmd.c_id);
2070         } else if (cmd.c_action == M_INITDEFAULT) {
2071             /*
2072             * initdefault is no longer meaningful, as the SMF
2073             * milestone controls what (legacy) run level we
2074             * boot to.
2075             */
2076             console(B_TRUE,
2077                 "Ignoring legacy \"initdefault\" entry.\n");
2078         } else if (cmd.c_action == M_SYSINIT) {
2079             /*
2080             * Execute the "sysinit" entry and wait for it to

```

```

2081         * complete. No bookkeeping is performed on these
2082         * entries because we avoid writing to the file system
2083         * until after there has been a chance to check it.
2084         */
2085         if (process = findpslot(&cmd)) {
2086             (void) sighold(SIGCLD);
2087             (void) snprintf(svc_aux, SVC_AUX_SIZE,
2088                 INITTAB_ENTRY_ID_STR_FORMAT, cmd.c_id);
2089             (void) snprintf(init_svc_fmri, SVC_FMRI_SIZE,
2090                 SVC_INIT_PREFIX INITTAB_ENTRY_ID_STR_FORMAT,
2091                 cmd.c_id);
2092             if (legacy_tmpl >= 0) {
2093                 (void) ct_pr_tmpl_set_svc_fmri(
2094                     legacy_tmpl, init_svc_fmri);
2095                 (void) ct_pr_tmpl_set_svc_aux(
2096                     legacy_tmpl, svc_aux);
2097             }

2099             for (oprocess = process;
2100                 (process = efork(M_OFF, oprocess,
2101                     (NAMED|NOCLEANUP))) == NO_ROOM;
2102                 /* CSTYLED */)
2103                 ;
2104             (void) sigrelse(SIGCLD);

2106             if (process == NULLPROC) {
2107                 maxfiles = ulimit(UL_GDESLIM, 0);

2109                 for (i = 0; i < maxfiles; i++)
2110                     (void)fcntl(i, F_SETFD,
2111                         FD_CLOEXEC);
2112                 (void) execle(SH, "INITSH", "-c",
2113                     cmd.c_command,
2114                     (char *)0, glob_envp);
2115                 console(B_TRUE,
2116                     "Command\n\"%s\"\n failed to execute.  errno = %d (exec of shell failed)\n",
2117                     cmd.c_command, errno);
2118                 exit(1);
2119             } else
2120                 while (waitproc(process) == FAILURE)
2121                     ;
2122             } else while (waitproc(process) == FAILURE);
2123             process->p_flags = 0;
2124             st_write();
2125         }
2126     }

2128     /* Restore the path. */
2129     free(glob_envp[0]);
2130     glob_envp[0] = old_path;

2132     /*
2133     * This will enable st_write() to complain about init_state_file.
2134     */
2135     booting = 0;

2137     /*
2138     * If the /etc/ioctl.syscon didn't exist or had invalid contents write
2139     * out a correct version.
2140     */
2141     if (write_ioctl)
2142         write_ioctl_syscon();

2144     /*
2145     * Start svc.startd(1M), which does most of the work.

```

```

2146  */
2147  if (startd_cline[0] != '\0' && startd_tmpl >= 0) {
2148      /* Start svc.startd. */
2149      if (startd_run(startd_cline, startd_tmpl, 0) == -1)
2150          cur_state = SINGLE_USER;
2151  } else {
2152      console(B_TRUE, "Absent svc.startd entry or bad "
2153              "contract template. Not starting svc.startd.\n");
2154      enter_maintenance();
2155  }
2156  }
  _____
  unchanged_portion_omitted_
  _____
2811 /*
2812  * prog_name() searches for the word or unix path name and
2813  * returns a pointer to the last element of the pathname.
2814  */
2815 static char *
2816 prog_name(char *string)
2817 {
2818     char *ptr, *ptr2;
2819     static char word[UT_USER_SZ + 1];
2820     /* XXX - utmp - fix name length */
2821     static char word[_POSIX_LOGIN_NAME_MAX];
2822
2823     /*
2824     * Search for the first word skipping leading spaces and tabs.
2825     */
2826     while (*string == ' ' || *string == '\t')
2827         string++;
2828
2829     /*
2830     * If the first non-space non-tab character is not one allowed in
2831     * a word, return a pointer to a null string, otherwise parse the
2832     * pathname.
2833     */
2834     if (*string != '.' && *string != '/' && *string != '_' &&
2835         (*string < 'a' || *string > 'z') &&
2836         (*string < 'A' || *string > 'Z') &&
2837         (*string < '0' || *string > '9'))
2838         return ("");
2839
2840     /*
2841     * Parse the pathname looking forward for '/', '\t', '\n' or
2842     * '\0'. Each time a '/' is found, move "ptr" to one past the
2843     * '/', thus when a '\t', '\n', or '\0' is found, "ptr" will
2844     * point to the last element of the pathname.
2845     */
2846     for (ptr = string; *string != ' ' && *string != '\t' &&
2847          *string != '\n' && *string != '\0'; string++) {
2848         if (*string == '/')
2849             ptr = string+1;
2850     }
2851
2852     /*
2853     * Copy out up to the size of the "ut_user" array into "word",
2854     * null terminate it and return a pointer to it.
2855     */
2856     for (ptr2 = &word[0]; ptr2 < &word[UT_USER_SZ] &&
2857          /* XXX - utmp - fix name length */
2858          for (ptr2 = &word[0]; ptr2 < &word[_POSIX_LOGIN_NAME_MAX - 1] &&
2859               ptr < string; /* CSTYLE */
2860               *ptr2++ = *ptr++);
2861
2862     *ptr2 = '\0';
2863     return (&word[0]);

```

```

2860  }
  _____
  unchanged_portion_omitted_
  _____
3791 /*
3792  * /etc/inittab has more entries and we have run out of room in the proc_table
3793  * array. Double the size of proc_table to accomodate the extra entries.
3794  */
3795 static void
3796 increase_proc_table_size()
3797 {
3798     sigset_t block, unblock;
3799     void *ptr;
3800     size_t delta = num_proc * sizeof (struct PROC_TABLE);
3801
3802     /*
3803     * Block signals for realloc.
3804     */
3805     (void) sigfillset(&block);
3806     (void) sigprocmask(SIG_BLOCK, &block, &unblock);
3807
3808     /*
3809     * On failure we just return because callers of this function check
3810     * for failure.
3811     */
3812     do
3813         ptr = realloc(g_state, g_state_sz + delta);
3814     while (ptr == NULL && errno == EAGAIN);
3815
3816     if (ptr != NULL) {
3817         /* ensure that the new part is initialized to zero */
3818         bzero((caddr_t)ptr + g_state_sz, delta);
3819
3820         g_state = ptr;
3821         g_state_sz += delta;
3822         num_proc <= 1;
3823     }
3824
3825     /* unblock our signals before returning */
3826     (void) sigprocmask(SIG_SETMASK, &unblock, NULL);
3827 }
  _____
  unchanged_portion_omitted_
  _____
3880 /*
3881  * Initialize our state.
3882  */
3883 * If the system just booted, then init_state_file, which is located on an
3884 * everpresent tmpfs filesystem, should not exist.
3885 *
3886 * If we were restarted, then init_state_file should exist, in
3887 * which case we'll read it in, sanity check it, and use it.
3888 *
3889 * Note: You can't call console() until proc_table is ready.
3890 */
3891 void
3892 st_init()
3893 {
3894     struct stat stb;
3895     int ret, st_fd, insane = 0;
3896     size_t to_be_read;
3897     char *ptr;

```



```

4092         return (-1);
4093     }

4095     if (err = ct_pr_tmpl_set_param(fd, CT_PR_INHERIT | CT_PR_REGENT))
4096         console(B_TRUE, "Contract set template inherit, regent "
4097             "failed: %s.\n", strerror(err));

4099     /*
4100      * These errors result in a misconfigured template, which is better
4101      * than no template at all, so warn but don't abort.
4102      */
4103     if (err = ct_tmpl_set_informative(fd, info))
4104         console(B_TRUE, ioctl_tset_emsg, "informative", strerror(err));

4106     if (err = ct_tmpl_set_critical(fd, critical))
4107         console(B_TRUE, ioctl_tset_emsg, "critical", strerror(err));

4109     if (err = ct_pr_tmpl_set_fatal(fd, fatal))
4110         console(B_TRUE, ioctl_tset_emsg, "fatal", strerror(err));

4112     if (err = ct_tmpl_set_cookie(fd, cookie))
4113         console(B_TRUE, ioctl_tset_emsg, "cookie", strerror(err));

4115     (void) fcntl(fd, F_SETFD, FD_CLOEXEC);

4117     return (fd);
4118 }

4120 /*
4121  * Create the templates and open an event file descriptor. We use dup2(2) to
4122  * get these descriptors away from the stdin/stdout/stderr group.
4123  */
4124 static void
4125 contracts_init()
4126 {
4127     int err, fd;

4129     /*
4130      * Create & configure a legacy template. We only want empty events so
4131      * we know when to abandon them.
4132      */
4133     legacy_tmpl = contract_make_template(0, CT_PR_EV_EMPTY, CT_PR_EV_HWERR,
4134         ORDINARY_COOKIE);
4135     if (legacy_tmpl >= 0) {
4136         err = ct_tmpl_activate(legacy_tmpl);
4137         if (err != 0) {
4138             (void) close(legacy_tmpl);
4139             legacy_tmpl = -1;
4140             console(B_TRUE,
4141                 "Couldn't activate legacy template (%s): "
4142                 "legacy services will be in init's contract.\n",
4143                 strerror(err));
4144         }
4145     } else
4146         console(B_TRUE,
4147             "Legacy services will be in init's contract.\n");

4149     if (dup2(legacy_tmpl, 255) == -1) {
4150         console(B_TRUE, "Could not duplicate legacy template: %s.\n",
4151             strerror(errno));
4152     } else {
4153         (void) close(legacy_tmpl);
4154         legacy_tmpl = 255;
4155     }

4157     (void) fcntl(legacy_tmpl, F_SETFD, FD_CLOEXEC);

```

```

4159     startd_tmpl = contract_make_template(0, CT_PR_EV_EMPTY,
4160         CT_PR_EV_HWERR | CT_PR_EV_SIGNAL | CT_PR_EV_CORE, STARTD_COOKIE);

4162     if (dup2(startd_tmpl, 254) == -1) {
4163         console(B_TRUE, "Could not duplicate startd template: %s.\n",
4164             strerror(errno));
4165     } else {
4166         (void) close(startd_tmpl);
4167         startd_tmpl = 254;
4168     }

4170     (void) fcntl(startd_tmpl, F_SETFD, FD_CLOEXEC);

4172     if (legacy_tmpl < 0 && startd_tmpl < 0) {
4173         /* The creation errors have already been reported. */
4174         console(B_TRUE,
4175             "Ignoring contract events. Core smf(5) services will not "
4176             "be restarted.\n");
4177         return;
4178     }

4180     /*
4181      * Open an event endpoint.
4182      */
4183     do
4184         fd = open64(CTFS_ROOT "/process/pbundle", O_RDONLY);
4185     while (fd < 0 && errno == EINTR);
4186     ;
4187     while (fd < 0 && errno == EINTR);
4188     if (fd < 0) {
4189         console(B_TRUE,
4190             "Couldn't open process pbundle: %s. Core smf(5) services "
4191             "will not be restarted.\n", strerror(errno));
4192         return;
4193     }

4194     if (dup2(fd, 253) == -1) {
4195         console(B_TRUE, "Could not duplicate process bundle: %s.\n",
4196             strerror(errno));
4197     } else {
4198         (void) close(fd);
4199         fd = 253;
4200     }

4202     (void) fcntl(fd, F_SETFD, FD_CLOEXEC);

4204     /* Reset in case we've been restarted. */
4205     (void) ct_event_reset(fd);

4207     poll_fds[0].fd = fd;
4208     poll_fds[0].events = POLLIN;
4209     poll_nfds = 1;
4210 }

4212 static int
4213 contract_getfile(ctid_t id, const char *name, int oflag)
4214 {
4215     int fd;

4217     do
4218         fd = contract_open(id, "process", name, oflag);
4219     while (fd < 0 && errno == EINTR);
4220     ;
4221     while (fd < 0 && errno == EINTR);

```



new/usr/src/cmd/init/init.c

17

```
4222     if (fd < 0)
4223         console(B_TRUE, "Couldn't open %s for contract %ld: %s.\n",
4224                 name, id, strerror(errno));
4226     return (fd);
4227 }
```

unchanged\_portion\_omitted

```

*****
13232 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/last/last.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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, Version 1.0 only
6  * (the "License").  You may not use this file except in compliance
7  * with the License.
8  *
9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 * or http://www.opensolaris.org/os/licensing.
11 * See the License for the specific language governing permissions
12 * and limitations under the License.
13 *
14 * When distributing Covered Code, include this CDDL HEADER in each
15 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 * If applicable, add the following below this CDDL HEADER, with the
17 * fields enclosed by brackets "[]" replaced with your own identifying
18 * information: Portions Copyright [yyyy] [name of copyright owner]
19 *
20 * CDDL HEADER END
21 */
22 /*
23 * Copyright (c) 2013 Gary Mills
24 *
25 * Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
26 * Use is subject to license terms.
27 */
28
29 /*
30 *      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T
31 *      All Rights Reserved
32 */
33
34 /*
35 * University Copyright- Copyright (c) 1982, 1986, 1988
36 * The Regents of the University of California
37 * All Rights Reserved
38 *
39 * University Acknowledgment- Portions of this document are derived from
40 * software developed by the University of California, Berkeley, and its
41 * contributors.
42 */
43
44 #pragma ident      "%Z%M% %I%      %E% SMI"
45
46 /*
47 * last
48 */
49 #include <sys/types.h>
50 #include <stdio.h>
51 #include <stdlib.h>
52 #include <unistd.h>
53 #include <strings.h>
54 #include <signal.h>
55 #include <sys/stat.h>
56 #include <pwd.h>
57 #include <fcntl.h>
58 #include <utmpx.h>
59 #include <locale.h>
60 #include <ctype.h>

```

```

60 /*
61  * Use the full lengths from utmpx for NMAX, LMAX and HMAX .
62  * NMAX, LMAX and HMAX are set to these values for now. They
63  * should be much higher because of the max allowed limit in
64  * utmpx.h
65 */
66 #define NMAX      (sizeof (((struct utmpx *)0)->ut_user))
67 #define LMAX      (sizeof (((struct utmpx *)0)->ut_line))
68 #define NMAX      8
69 #define LMAX      12
70 #define HMAX      (sizeof (((struct utmpx *)0)->ut_host))
71
72 /* Print minimum field widths. */
73 #define LOGIN_WIDTH 8
74 #define LINE_WIDTH  12
75
76 #define SECDAY      (24*60*60)
77 #define CHUNK_SIZE 256
78
79 #define lineq(a, b)      (strncmp(a, b, LMAX) == 0)
80 #define nameq(a, b)     (strncmp(a, b, NMAX) == 0)
81 #define hosteq(a, b)    (strncmp(a, b, HMAX) == 0)
82 #define linehostnameq(a, b, c, d) \
83     (lineq(a, b)&&hosteq(a+LMAX+1, c)&&nameq(a+LMAX+HMAX+2, d))
84
85 #define USAGE      "usage: last [-n number] [-f filename] [-a ] [name | tty] ...\n"
86
87 /* Beware: These are set in main() to exclude the executable name. */
88 static char      **argv;
89 static int        argc;
90 static char      **names;
91 static int        names_num;
92
93 static struct      utmpx buf[128];
94
95 /*
96  * ttnames and logouts are allocated in the blocks of
97  * CHUNK_SIZE lines whenever needed. The count of the
98  * current size is maintained in the variable "lines"
99  * The variable bootxtime is used to hold the time of
100 * the last BOOT_TIME
101 * All elements of the logouts are initialised to bootxtime
102 * everytime the buffer is reallocated.
103 */
104 static char      **ttnames;
105 static time_t    *logouts;
106 static time_t    bootxtime;
107 static int        lines;
108 static char      timef[128];
109 static char      hostf[HMAX + 1];
110
111 static char *strspl(char *, char *);
112 static void onintr(int);
113 static void reallocate_buffer();
114 static void memory_alloc(int);
115 static int want(struct utmpx *, char **, char **);
116 static void record_time(time_t *, int *, int, struct utmpx *);
117
118 int
119 main(int ac, char **av)
120 {
121     int i, j;
122     int aflag = 0;
123     int fpos;      /* current position in time format buffer */

```

```

120     int chrcnt;      /* # of chars formatted by current sprintf */
121     int bl, wtmp;
122     char *ct;
123     char *ut_host;
124     char *ut_user;
125     struct utmpx *bp;
126     time_t otime;
127     struct stat stb;
128     int print = 0;
129     char *crmsg = (char *)0;
130     long outrec = 0;
131     long maxrec = 0x7fffffffL;
132     char *wtmpfile = "/var/adm/wtmpx";
133     size_t hostf_len;

135     (void) setlocale(LC_ALL, "");
136 #if !defined(TEXT_DOMAIN) /* Should be defined by cc -D */
137 #define TEXT_DOMAIN "SYS_TEST" /* Use this only if it weren't. */
138 #endif
139     (void) textdomain(TEXT_DOMAIN);

141     (void) time(&buf[0].ut_xtime);
142     ac--, av++;
143     argc = ac;
144     argv = av;
145     names = malloc(argc * sizeof (char *));
146     if (names == NULL) {
147         perror("last");
148         exit(2);
149     }
150     names_num = 0;
151     for (i = 0; i < argc; i++) {
152         if (argv[i][0] == '-') {

154             /* -[0-9]* sets max # records to print */
155             if (isdigit(argv[i][1])) {
156                 maxrec = atoi(argv[i]+1);
157                 continue;
158             }

160             for (j = 1; argv[i][j] != '\0'; ++j) {
161                 switch (argv[i][j]) {

163                     /* -f name sets filename of wtmp file */
164                     case 'f':
165                         if (argv[i][j+1] != '\0') {
166                             wtmpfile = &argv[i][j+1];
167                         } else if (i+1 < argc) {
168                             wtmpfile = argv[i+1];
169                         } else {
170                             (void) fprintf(stderr,
171                                 gettext("last: argument to "
172                                     "-f is missing\n"));
173                             (void) fprintf(stderr,
174                                 gettext(USAGE));
175                             exit(1);
176                         }
177                         goto next_word;

179                     /* -n number sets max # records to print */
180                     case 'n': {
181                         char *arg;

183                         if (argv[i][j+1] != '\0') {
184                             arg = &argv[i][j+1];
185                         } else if (i+1 < argc) {

```

```

186             arg = argv[++i];
187         } else {
188             (void) fprintf(stderr,
189                 gettext("last: argument to "
190                     "-n is missing\n"));
191             (void) fprintf(stderr,
192                 gettext(USAGE));
193             exit(1);
194         }

196         if (!isdigit(*arg)) {
197             (void) fprintf(stderr,
198                 gettext("last: argument to "
199                     "-n is not a number\n"));
200             (void) fprintf(stderr,
201                 gettext(USAGE));
202             exit(1);
203         }
204         maxrec = atoi(arg);
205         goto next_word;
206     }

208     /* -a displays hostname last on the line */
209     case 'a':
210         aflag++;
211         break;

213     default:
214         (void) fprintf(stderr, gettext(USAGE));
215         exit(1);
216     }
217 }

219 next_word:
220     continue;
221 }

223     if (strlen(argv[i]) > 2 || strcmp(argv[i], "~") == 0 ||
224         getpwnam(argv[i]) != NULL) {
225         /* Not a tty number. */
226         names[names_num] = argv[i];
227         ++names_num;
228     } else {
229         /* tty number. Prepend "tty". */
230         names[names_num] = strspn("tty", argv[i]);
231         ++names_num;
232     }
233 }

235     wtmp = open(wtmpfile, 0);
236     if (wtmp < 0) {
237         perror(wtmpfile);
238         exit(1);
239     }
240     (void) fstat(wtmp, &stb);
241     bl = (stb.st_size + sizeof (buf)-1) / sizeof (buf);
242     if (signal(SIGINT, SIG_IGN) != SIG_IGN) {
243         (void) signal(SIGINT, onintr);
244         (void) signal(SIGQUIT, onintr);
245     }
246     lines = CHUNK_SIZE;
247     tttnames = calloc(lines, sizeof (char *));
248     logouts = calloc(lines, sizeof (time_t));
249     if (tttnames == NULL || logouts == NULL) {
250         (void) fprintf(stderr, gettext("Out of memory \n "));
251         exit(2);

```

```

252     }
253     for (bl--; bl >= 0; bl--) {
254         (void) lseek(wtmp, (off_t)(bl * sizeof (buf)), 0);
255         bp = &buf[read(wtmp, buf, sizeof (buf)) / sizeof (buf[0]) - 1];
256         for (; bp >= buf; bp--) {
257             if (want(bp, &ut_host, &ut_user)) {
258                 for (i = 0; i <= lines; i++) {
259                     if (i == lines)
260                         reallocate_buffer();
261                     if (ttnames[i] == NULL) {
262                         memory_alloc(i);
263                         /*
264                          * LMAX+HMAX+NMAX+3 bytes have been
265                          * allocated for ttnames[i].
266                          * If bp->ut_line is longer than LMAX,
267                          * ut_host is longer than HMAX,
268                          * and ut_user is longer than NMAX,
269                          * truncate it to fit ttnames[i].
270                          */
271                         (void) strncpy(ttnames[i], bp->ut_line,
272                                         LMAX+1);
273                         (void) strncpy(ttnames[i]+LMAX+1,
274                                         ut_host, HMAX+1);
275                         (void) strncpy(ttnames[i]+LMAX+HMAX+2,
276                                         ut_user, NMAX+1);
277                         record_time(&otime, &print,
278                                     i, bp);
279                         break;
280                     } else if (linehostnameq(ttnames[i],
281                                             bp->ut_line, ut_host, ut_user)) {
282                         record_time(&otime,
283                                     &print, i, bp);
284                         break;
285                     }
286                 }
287             }
288             if (print) {
289                 if (strcmp(bp->ut_line, "ftp", 3) == 0)
290                     bp->ut_line[3] = '\0';
291                 if (strcmp(bp->ut_line, "uucp", 4) == 0)
292                     bp->ut_line[4] = '\0';
293
294                 ct = ctime(&bp->ut_xtime);
295                 (void) printf(gettext("%-*.s %-*.s "),
296                               LOGIN_WIDTH, NMAX, bp->ut_name,
297                               LINE_WIDTH, LMAX, bp->ut_line);
298                 NMAX, NMAX, bp->ut_name,
299                 LMAX, LMAX, bp->ut_line);
300                 hostf_len = strlen(bp->ut_host);
301                 (void) snprintf(hostf, sizeof (hostf),
302                                 "%-*.s", hostf_len, hostf_len,
303                                 bp->ut_host);
304                 fpos = snprintf(timef, sizeof (timef),
305                                 "%10.10s %5.5s ",
306                                 ct, 11 + ct);
307                 if (!lineq(bp->ut_line, "system boot") &&
308                     !lineq(bp->ut_line, "system down")) {
309                     if (otime == 0 &&
310                         bp->ut_type == USER_PROCESS) {
311
312                     if (fpos < sizeof (timef)) {
313                         /* timef still has room */
314                         (void) snprintf(timef + fpos, sizeof (timef) - fpos,
315                                         gettext(" still logged in"));
316                     }

```

```

316     } else {
317         time_t delta;
318         if (otime < 0) {
319             otime = -otime;
320             /*
321              * TRANSLATION_NOTE
322              * See other notes on "down"
323              * and "- %5.5s".
324              * "-" means "until". This
325              * is displayed after the
326              * starting time as in:
327              * 16:20 - down
328              * You probably don't want to
329              * translate this. Should you
330              * decide to translate this,
331              * translate "- %5.5s" too.
332              */
333
334         if (fpos < sizeof (timef)) {
335             /* timef still has room */
336             chrnt = snprintf(timef + fpos, sizeof (timef) - fpos,
337                             gettext("- %s"), crmsg);
338             fpos += chrnt;
339         }
340
341         } else {
342
343         if (fpos < sizeof (timef)) {
344             /* timef still has room */
345             chrnt = snprintf(timef + fpos, sizeof (timef) - fpos,
346                             gettext("- %5.5s"), ctime(&otime) + 11);
347             fpos += chrnt;
348         }
349
350         }
351         delta = otime - bp->ut_xtime;
352         if (delta < SECDAY) {
353
354         if (fpos < sizeof (timef)) {
355             /* timef still has room */
356             (void) snprintf(timef + fpos, sizeof (timef) - fpos,
357                             gettext(" (%5.5s)", asctime(gmtime(&delta)) + 11);
358         }
359
360         } else {
361
362         if (fpos < sizeof (timef)) {
363             /* timef still has room */
364             (void) snprintf(timef + fpos, sizeof (timef) - fpos,
365                             gettext(" (%ld+%5.5s)", delta / SECDAY,
366                                     asctime(gmtime(&delta)) + 11);
367         }
368
369         }
370
371         }
372         if (aflag)
373             (void) printf("%-35.35s %-*.s\n",
374                           timef, strlen(hostf), hostf);
375         else
376             (void) printf("%-16.16s %-*.35s\n",
377                           hostf, timef);
378         (void) fflush(stdout);
379         if (++outrec >= maxrec)
380             exit(0);
381     }

```

```
382      /*
383      * when the system is down or crashed.
384      */
385      if (bp->ut_type == BOOT_TIME) {
386          for (i = 0; i < lines; i++)
387              logouts[i] = -bp->ut_xtime;
388          bootxtime = -bp->ut_xtime;
389          /*
390          * TRANSLATION_NOTE
391          * Translation of this "down " will replace
392          * the %s in "- %s". "down" is used instead
393          * of the real time session was ended, probably
394          * because the session ended by a sudden crash.
395          */
396          crmsg = gettext("down ");
397      }
398      print = 0;      /* reset the print flag */
399  }
400  }
401  ct = ctime(&buf[0].ut_xtime);
402  (void) printf(gettext("\nwtmp begins %10.10s %5.5s \n"), ct, ct + 11);
404  /* free() called to prevent lint warning about names */
405  free(names);
407  return (0);
408 }
_____unchanged_portion_omitted_____
```

```

*****
21455 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/newtask/newtask.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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, Version 1.0 only
6  * (the "License").  You may not use this file except in compliance
7  * with the License.
8  *
9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 * or http://www.opensolaris.org/os/licensing.
11 * See the License for the specific language governing permissions
12 * and limitations under the License.
13 *
14 * When distributing Covered Code, include this CDDL HEADER in each
15 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 * If applicable, add the following below this CDDL HEADER, with the
17 * fields enclosed by brackets "[]" replaced with your own identifying
18 * information: Portions Copyright [yyyy] [name of copyright owner]
19 *
20 * CDDL HEADER END
21 */
22 /*
23 * Copyright (c) 2013 Gary Mills
24 *
25 * Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
26 * Use is subject to license terms.
27 */

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

29 #include <sys/types.h>
30 #include <sys/task.h>

32 #include <alloca.h>
33 #include <libproc.h>
34 #include <libintl.h>
35 #include <libgen.h>
36 #include <limits.h>
37 #include <project.h>
38 #include <pwd.h>
39 #include <secdb.h>
40 #include <stdio.h>
41 #include <stdlib.h>
42 #include <string.h>
43 #include <sys/varargs.h>
44 #include <unistd.h>
45 #include <errno.h>
46 #include <signal.h>
47 #include <priv_utils.h>

49 #include "utils.h"

51 #define OPTIONS_STRING  "Fc:lp:v"
52 #define NENV            8
53 #define ENVSIZE        255
54 #define PATH            "PATH=/usr/bin"
55 #define SUPATH         "PATH=/usr/sbin:/usr/bin"
56 #define SHELL           "/usr/bin/sh"
57 #define SHELL2         "/sbin/sh"
58 #define TIMEZONEFILE   "/etc/default/init"

```

```

59 #define LOGINFILE      "/etc/default/login"
60 #define GLOBAL_ERR_SZ  1024
61 #define GRAB_RETRY_MAX 100

63 static const char *pname;
64 extern char **environ;
65 static char *supath = SUPATH;
66 static char *path = PATH;
67 static char global_error[GLOBAL_ERR_SZ];
68 static int verbose = 0;

70 static priv_set_t *nset;

72 /* Private definitions for libproject */
73 extern projid_t setproject_proc(const char *, const char *, int, pid_t,
74     struct ps_prochandle *, struct project *);
75 extern priv_set_t *setproject_initpriv(void);

77 static void usage(void);

79 static void preserve_error(const char *format, ...);

81 static int update_running_proc(int, char *, char *);
82 static int set_ids(struct ps_prochandle *, struct project *,
83     struct passwd *);
84 static struct passwd *match_user(uid_t, char *, int);
85 static void setproject_err(char *, char *, int, struct project *);

87 static void
88 usage(void)
89 {
90     (void) fprintf(stderr, gettext("usage: \n\t%s [-v] [-p project] "
91     "[-c pid] [-F1] [command [args ...]]\n"), pname);
92     exit(2);
93 }

unchanged portion omitted

649 /*
650 * Given the input arguments, return the passwd structure that matches best.
651 * Also, since we use getpwnam() and friends, subsequent calls to this
652 * function will re-use the memory previously returned.
653 */
654 static struct passwd *
655 match_user(uid_t uid, char *projname, int is_my_uid)
656 {
657     char prbuf[PROJECT_BUFSZ], username[LOGNAME_MAX+1];
658     struct project prj;
659     char *tmp_name;
660     struct passwd *pw = NULL;

662     /*
663     * In order to allow users with the same UID but distinguishable
664     * user names to be in different projects we play a guessing
665     * game of which username is most appropriate.  If we're checking
666     * for the uid of the calling process, the login name is a
667     * good starting point.
668     */
669     if (is_my_uid) {
670         if ((tmp_name = getlogin()) == NULL ||
671             (pw = getpwnam(tmp_name)) == NULL || (pw->pw_uid != uid) ||
672             (pw->pw_name == NULL))
673             pw = NULL;
674     }

676     /*
677     * If the login name doesn't work, we try the first match for

```

```
678     * the current uid in the password file.
679     */
680     if (pw == NULL) {
681         if ((pw = getpwuid(uid)) == NULL) || pw->pw_name == NULL) {
682             preserve_error(gettext("cannot find username "
683                 "for uid %d"), uid);
684             return (NULL);
685         }
686     }
687
688     /*
689     * If projname wasn't supplied, we've done our best, so just return
690     * what we've got now. Alternatively, if newtask's invoker has
691     * superuser privileges, return the pw structure we've got now, with
692     * no further checking from inproj(). Superuser should be able to
693     * join any project, and the subsequent call to setproject() will
694     * allow this.
695     */
696     if (projname == NULL || getuid() == (uid_t)0)
697         return (pw);
698
699     (void) strncpy(username, pw->pw_name, sizeof (username));
700     (void) strcpy(pw_name, pw->pw_name);
701
702     if (inproj(username, projname, prbuf, PROJECT_BUFSZ) == 0) {
703         char **u;
704         tmp_name = NULL;
705
706         /*
707         * If the previous guesses didn't work, walk through all
708         * project members and test for UID-equivalence.
709         */
710
711         if (getprojbyname(projname, &prj, prbuf,
712             PROJECT_BUFSZ) == NULL) {
713             preserve_error(gettext("unknown project \"%s\""),
714                 projname);
715             return (NULL);
716         }
717
718         for (u = prj.pj_users; *u; u++) {
719             if ((pw = getpwnam(*u)) == NULL)
720                 continue;
721
722             if (pw->pw_uid == uid) {
723                 tmp_name = pw->pw_name;
724                 break;
725             }
726         }
727
728         if (tmp_name == NULL) {
729             preserve_error(gettext("user \"%s\" is not a member of "
730                 "project \"%s\""), username, projname);
731             return (NULL);
732         }
733     }
734
735     return (pw);
736 }
737
738 unchanged_portion_omitted
```

new/usr/src/cmd/oamuser/inc/users.h

1

```
*****
2641 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/inc/users.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
25  */

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

31 #ifndef _USERS_H
32 #define _USERS_H

35 #include <pwd.h>
36 #include <grp.h>
37 #include <project.h>

39 #define GROUP          "/etc/group"

41 /* max number of projects that can be specified when adding a user */
42 #define NPROJECTS_MAX  1024

44 /* validation returns */
45 #define NOTUNIQUE      0      /* not unique */
46 #define RESERVED      1      /* reserved */
47 #define UNIQUE        2      /* is unique */
48 #define TOOBIG        3      /* number too big */
49 #define INVALID       4
50 #define LONGNAME      5      /* string too long */

52 /*
53  * Note: constraints checking for warning (release 2.6),
54  * and these may be enforced in the future releases.
55  */
56 #define WARN_NAME_TOO_LONG      0x1
57 #define WARN_BAD_GROUP_NAME     0x2
58 #define WARN_BAD_LOGNAME_CHAR  0x4
59 #define WARN_BAD_LOGNAME_FIRST 0x8
60 #define WARN_NO_LOWERCHAR       0x10
```

new/usr/src/cmd/oamuser/inc/users.h

2

```
61 #define WARN_BAD_PROJ_NAME     0x20
62 #define WARN_LOGGED_IN        0x40

64 /* Exit codes from passmgmt */
65 #define PEX_SUCCESS            0
66 #define PEX_NO_PERM           1
67 #define PEX_SYNTAX            2
68 #define PEX_BADARG            3
69 #define PEX_BADUID            4
70 #define PEX_HOSED_FILES       5
71 #define PEX_FAILED            6
72 #define PEX_MISSING           7
73 #define PEX_BUSY              8
74 #define PEX_BADNAME           9

76 #define REL_PATH(x)           (x && *x != '/')

78 /*
79  * interfaces available from the library
80  */
81 extern int valid_login(char *, struct passwd **, int *);
82 extern int valid_gname(char *, struct group **, int *);
83 extern int valid_group(char *, struct group **, int *);
84 extern int valid_project(char *, struct project *, void *buf, size_t, int *);
85 extern int valid_projname(char *, struct project *, void *buf, size_t, int *);
86 extern void warningmsg(int, char *);
87 extern void putgrent(struct group *, FILE *);

89 /* passmgmt */
90 #define PASSMGMT              "/usr/lib/passmgmt";
91 #endif /* _USERS_H */
```



```

*****
2682 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/lib/vlogin.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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, Version 1.0 only
6  * (the "License").  You may not use this file except in compliance
7  * with the License.
8  *
9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 * or http://www.opensolaris.org/os/licensing.
11 * See the License for the specific language governing permissions
12 * and limitations under the License.
13 *
14 * When distributing Covered Code, include this CDDL HEADER in each
15 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 * If applicable, add the following below this CDDL HEADER, with the
17 * fields enclosed by brackets "[]" replaced with your own identifying
18 * information: Portions Copyright [yyyy] [name of copyright owner]
19 *
20 * CDDL HEADER END
21 */
22 /*
23 * Copyright (c) 2013 Gary Mills
24 *
25 * Copyright (c) 1997, by Sun Microsystems, Inc.
26 * All rights reserved.
27 */

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

31 #pragma ident      "%Z%M% %I%      %E% SMI"      /* SVr4.0 1.3 */

32 /*LINTLIBRARY*/

34 #include      <sys/types.h>
35 #include      <stdio.h>
36 #include      <ctype.h>
37 #include      <userdefs.h>
38 #include      <users.h>
39 #include      <deflt.h>
40 #include      <limits.h>

42 /* Defaults file */
43 #define DEFAULT_USERADD "/etc/default/useradd"

45 /*
46  * validate string given as login name.
47  */
48 int
49 valid_login(char *login, struct passwd **pptr, int *warning)
50 {
51     struct passwd *t_pptr;
52     char *ptr = login;
53     int badlchar, badc, clower, len;
54     char c;
55     char action;

57     len = 0; clower = 0; badc = 0; badlchar = 0;

```

```

58     *warning = 0;
59     if (!login || !*login)
60         return (INVALID);

62     c = *ptr;
63     if (!isalpha(c))
64         badlchar++;
65     for (; c != NULL; ptr++, c = *ptr) {
66         len++;
67         if (!isprint(c) || (c == ':') || (c == '\n'))
68             return (INVALID);
69         if (!isalnum(c) && c != '_' && c != '-' && c != '.')
70             badc++;
71         if (islower(c))
72             clower++;
73     }

75     action = 'w';
76     if (defopen(DEFAULT_USERADD) == 0) {
77         char *defptr;

79         if ((defptr = defread("EXCEED_TRAD=")) != NULL) {
80             char let = tolower(*defptr);

82             switch (let) {
83                 case 'w':      /* warning */
84                 case 'e':      /* error */
85                 case 's':      /* silent */
86                     action = let;
87                     break;
88             }
89         }
90         (void) defopen((char *)NULL);
91     }

93     /*
94     * XXX length checking causes some operational/compatibility problem.
95     * This has to be revisited in the future as ARC/standards issue.
96     */
97     if (len > LOGNAME_MAX)
98         return (LONGNAME);

100     if (len > LOGNAME_MAX_TRAD) {
101         if (action == 'w')
102             *warning = *warning | WARN_NAME_TOO_LONG;
103         else if (action == 'e')
104             return (LONGNAME);
105     }

107     if (clower == 0)
108         *warning = *warning | WARN_NO_LOWERCHAR;
109     if (badc != 0)
110         *warning = *warning | WARN_BAD_LOGNAME_CHAR;
111     if (badlchar != 0)
112         *warning = *warning | WARN_BAD_LOGNAME_FIRST;

114     if ((t_pptr = getpwnam(login)) != NULL) {
115         if (pptr) *pptr = t_pptr;
116         return (NOTUNIQUE);
117     }
118     return (UNIQUE);
119 }

```

\_\_\_\_\_unchanged\_portion\_omitted\_\_\_\_\_

```

*****
3238 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/user/Makefile
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 # Copyright (c) 2013 Gary Mills
22 #
23 # Copyright (c) 1990, 2010, Oracle and/or its affiliates. All rights reserved.
24 #
25 # cmd/oamuser/user/Makefile
26 #

28 DEFAULTFILES= useradd.dfl

30 include ../../Makefile.cmd

32 GREP=      grep

34 USERADD=   useradd
35 USERDEL=   userdel
36 USERMOD=   usermod
37 ROLEADD=   roleadd
38 ROLEDEL=   roledel
39 ROLEMOD=   rolemod

41 SBINPROG=  $(USERADD) $(USERDEL) $(USERMOD)
42 #
43 # Removing sysadm: deleted $(SYSADMPROG) from this target.
44 #
45 PROG=      $(SBINPROG)
46 PRODUCT=   $(PROG)

48 ADD_OBJ=   useradd.o      uid.o      homedir.o \
49            groups.o      call_pass.o  userdefs.o  messages.o \
50            val_lgrp.o    funcs.o     val_lprj.o  proj.o

52 DEL_OBJ=   userdel.o      call_pass.o  rmfiles.o   isbusy.o \
53            groups.o      messages.o   funcs.o     proj.o

55 MOD_OBJ=   usermod.o      uid.o      movedir.o   groups.o \
56            rmfiles.o    call_pass.o  isbusy.o    homedir.o \
57            userdefs.o   messages.o   val_lgrp.o  funcs.o \
58            val_lprj.o   proj.o

60 OBJECTS=   $(ADD_OBJ)      $(DEL_OBJ)  $(MOD_OBJ)

```

```

62 SRCS=      $(OBJECTS:.o=.c)

64 LIBDIR=    ../lib
65 LIBUSRGRP= $(LIBDIR)/lib.a
66 LIBADM=    -ladm
67 LOCAL=     ../inc
68 HERE=      .
69 LINTFLAGS= -u

71 ROOTSKEL=  $(ROOTETC)/skel
72 INSSBINPROG=$(SBINPROG:%=$(ROOTUSRSBIN)/%)
73 INSSKELFILE=$(SKELFILE:%=$(ROOTSKEL)/%)

75 CPPFLAGS=  -I$(HERE) -I$(LOCAL) $(CPPFLAGS.master)
76 CERRWARN += -_gcc=-Wno-implicit-function-declaration
77 CERRWARN += -_gcc=-Wno-type-limits
78 CERRWARN += -_gcc=-Wno-uninitialized
79 CERRWARN += -_gcc=-Wno-parentheses

81 $(INSSBINPROG) := FILEMODE = 0555
82 $(INSSYSADMPROG) := FILEMODE = 0500
83 $(INSSKELFILE) := FILEMODE = 0644

85 $(USERADD) := OBJS = $(ADD_OBJ)
86 $(USERADD) := LIBS = $(LIBUSRGRP)

88 $(USERDEL) := OBJS = $(DEL_OBJ)
89 $(USERDEL) := LIBS = $(LIBUSRGRP)

91 $(USERMOD) := OBJS = $(MOD_OBJ)
92 $(USERMOD) := LIBS = $(LIBUSRGRP)

94 LDLIBS +=  -lbsd -lnsl -lsecdb -lproject -ltsol

96 .PARALLEL: $(OBJECTS)

98 all:       $(PRODUCT)

100 $(PROG):   $$$(OBJ) $$$(LIBS)
101            $(LINK.c) $(OBJ) -o $$@ $(LIBS) $(LDLIBS)
102            $(POST_PROCESS)

104 $(USERADD): $(ADD_OBJ)
105 $(USERMOD): $(MOD_OBJ)
106 $(USERDEL): $(DEL_OBJ)

108 install:   all $(ROOTETCDEFAULTFILES) .WAIT \
109            $(ROOTSKEL) $(INSSBINPROG) $(INSSKELFILE)
105 install:   all .WAIT $(ROOTSKEL) $(INSSBINPROG) $(INSSKELFILE)
110            $(RM) $(ROOTUSRSBIN)/$(ROLEADD)
111            $(LN) $(ROOTUSRSBIN)/$(USERADD) $(ROOTUSRSBIN)/$(ROLEADD)
112            $(RM) $(ROOTUSRSBIN)/$(ROLEDEL)
113            $(LN) $(ROOTUSRSBIN)/$(USERDEL) $(ROOTUSRSBIN)/$(ROLEDEL)
114            $(RM) $(ROOTUSRSBIN)/$(ROLEMOD)
115            $(LN) $(ROOTUSRSBIN)/$(USERMOD) $(ROOTUSRSBIN)/$(ROLEMOD)

117 clean:
118            $(RM) $(OBJECTS)

120 lint:      lint_SRCS

122 include ../../Makefile.targ

```

```

*****
4796 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/user/messages.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 /*      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T */
22 /*      All Rights Reserved */

25 /*
26  * Copyright (c) 2013 Gary Mills
27  *
28  * Copyright 2006 Sun Microsystems, Inc. All rights reserved.
29  * Use is subject to license terms.
30  */

30 #pragma ident      "%Z%M% %I%      %E% SMI"      /* SVr4.0 1.6 */

32 char *errmsgs[] = {
33     "WARNING: uid %ld is reserved.\n",
34     "WARNING: more than NGROUPS_MAX(%d) groups specified.\n",
35     "ERROR: invalid syntax.\n"
36     "usage:  useradd [-u uid [-o] | -g group | -G group[,group]...] |"
37     "-d dir | -b base_dir |\n"
38     "\t\t-s shell | -c comment | -m [-k skel_dir] | -f inactive |\n"
39     "\t\t-e expire | -A authorization [, authorization ...] |\n"
40     "\t\t-P profile [, profile ...] | -R role [, role ...] |\n"
41     "\t\t-K key=value | -p project [, project ...] login\n"
42     "\tuseradd -D [-g group | -b base_dir | -f inactive | -e expire\n"
43     "\t\t-A authorization [, authorization ...] |\n"
44     "\t\t-P profile [, profile ...] | -R role [, role ...] |\n"
45     "\t\t-K key=value ... -p project] | [-s shell] | [-k skel_dir]\n",
46     "ERROR: Invalid syntax.\nusage:  userdel [-r] login\n",
47     "ERROR: Invalid syntax.\n"
48     "usage:  usermod -u uid [-o] | -g group | -G group[,group]... |\n"
49     "\t\t-d dir [-m] | -s shell | -c comment |\n"
50     "\t\t-l new_logname | -f inactive | -e expire |\n"
51     "\t\t-A authorization [, authorization ...] | -K key=value ... |\n"
52     "\t\t-P profile [, profile ...] | -R role [, role ...] login\n",
53     "ERROR: Unexpected failure. Defaults unchanged.\n",
54     "ERROR: Unable to remove files from home directory.\n",
55     "ERROR: Unable to remove home directory.\n",
56     "ERROR: Cannot update system files - login cannot be %s.\n",
57     "ERROR: uid %ld is already in use. Choose another.\n",
58     "ERROR: %s is already in use. Choose another.\n",

```

```

59     "ERROR: %s does not exist.\n",
60     "ERROR: %s is not a valid %s. Choose another.\n",
61     "ERROR: %s is in use. Cannot %s it.\n",
62     "WARNING: %s has no permissions to use %s.\n",
63     "ERROR: There is not sufficient space to move %s home directory to %s"
64     "\n",
65     "ERROR: %s %ld is too big. Choose another.\n",
66     "ERROR: group %s does not exist. Choose another.\n",
67     "ERROR: Unable to %s: %s.\n",
68     "ERROR: %s is not a full path name. Choose another.\n",
69     "ERROR: %s is the primary group name. Choose another.\n",
70     "ERROR: Inconsistent password files. See pwconv(1M).\n",
71     "ERROR: %s is not a local user.\n",
72     "ERROR: Permission denied.\n",
73     "WARNING: Group entry exceeds 2048 char: /etc/group entry truncated.\n",
74     "ERROR: invalid syntax.\n"
75     "usage:  roleadd [-u uid [-o] | -g group | -G group[,group]...] |"
76     "-d dir |\n"
77     "\t\t-s shell | -c comment | -m [-k skel_dir] | -f inactive |\n"
78     "\t\t-e expire | -A authorization [, authorization ...] |\n"
79     "\t\t-P profile [, profile ...] | -K key=value ] login\n"
80     "\troleadd -D [-g group | -b base_dir | -f inactive | -e expire\n"
81     "\t\t-A authorization [, authorization ...] |\n"
82     "\t\t-P profile [, profile ...] |\n",
83     "ERROR: Invalid syntax.\nusage:  roledel [-r] login\n",
84     "ERROR: Invalid syntax.\n"
85     "usage:  rolemod -u uid [-o] | -g group | -G group[,group]... |\n"
86     "\t\t-d dir [-m] | -s shell | -c comment |\n"
87     "\t\t-l new_logname | -f inactive | -e expire |\n"
88     "\t\t-A authorization [, authorization ...] | -K key=value |\n"
89     "\t\t-P profile [, profile ...] login\n",
90     "ERROR: project %s does not exist. Choose another.\n",
91     "WARNING: more than NPROJECTS_MAX(%d) projects specified.\n",
92     "WARNING: Project entry exceeds %d char: /etc/project entry truncated."
93     "\n",
94     "ERROR: Invalid key.\n",
95     "ERROR: Missing value specification.\n",
96     "ERROR: Multiple definitions of key ``%s``.\n",
97     "ERROR: Roles must be modified with ``rolemod``.\n",
98     "ERROR: Users must be modified with ``usermod``.\n",
99     "WARNING: gid %ld is reserved.\n",
100    "ERROR: Failed to read /etc/group file due to invalid entry or"
101    " read error.\n",
102    "ERROR: %s is too long. Choose another.\n",
103 };

```

unchanged portion omitted

new/usr/src/cmd/oamuser/user/messages.h

1

```
*****
4075 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/user/messages.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 /*      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T */
22 /*      All Rights Reserved */

25 /*
26  * Copyright (c) 2013 Gary Mills
27  *
28  * Copyright 2006 Sun Microsystems, Inc. All rights reserved.
29  * Use is subject to license terms.
30  */

32 #ifndef _MESSAGES_H
33 #define _MESSAGES_H

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

35 extern void errmsg(int, ...);

37 /* WARNING: uid %d is reserved. */
38 #define M_RESERVED      0

40 /* WARNING: more than NGROUPS_MAX(%d) groups specified. */
41 #define M_MAXGROUPS      1

43 /* ERROR: invalid syntax.\nusage: useradd ... */
44 #define M_AUSAGE      2

46 /* ERROR: Invalid syntax.\nusage: userdel [-r] login\n" */
47 #define M_DUSAGE      3

49 /* ERROR: Invalid syntax.\nusage: usermod ... */
50 #define M_MUSAGE      4

53 /* ERROR: Unexpected failure. Defaults unchanged. */
54 #define M_FAILED      5

56 /* ERROR: Unable to remove files from home directory. */
57 #define M_RMFILES      6
```

new/usr/src/cmd/oamuser/user/messages.h

2

```
59 /* ERROR: Unable to remove home directory. */
60 #define M_RMHOME      7

62 /* ERROR: Cannot update system files - login cannot be %s. */
63 #define M_UPDATE      8

65 /* ERROR: uid %d is already in use. Choose another. */
66 #define M_UID_USED      9

68 /* ERROR: %s is already in use. Choose another. */
69 #define M_USED      10

71 /* ERROR: %s does not exist. */
72 #define M_EXIST      11

74 /* ERROR: %s is not a valid %s. Choose another. */
75 #define M_INVALID      12

77 /* ERROR: %s is in use. Cannot %s it. */
78 #define M_BUSY      13

80 /* WARNING: %s has no permissions to use %s. */
81 #define M_NO_PERM      14

83 /* ERROR: There is not sufficient space to move %s home directory to %s */
84 #define M_NOSPACE      15

86 /* ERROR: %s %d is too big. Choose another. */
87 #define M_TOOBIG      16

89 /* ERROR: group %s does not exist. Choose another. */
90 #define M_GRP_NOTUSED      17

92 /* ERROR: Unable to %s: %s */
93 #define M_OOPS      18

95 /* ERROR: %s is not a full path name. Choose another. */
96 #define M_RELPATH      19

98 /* ERROR: %s is the primary group name. Choose another. */
99 #define M_SAME_GRP      20

101 /* ERROR: Inconsistent password files. See pwconv(1M). */
102 #define M_HOSSED_FILES      21

104 /* ERROR: %s is not a local user. */
105 #define M_NONLOCAL      22

107 /* ERROR: Permission denied. */
108 #define M_PERM_DENIED      23

110 /* WARNING: Group entry exceeds 2048 char: /etc/group entry truncated. */
111 #define M_GROUP_ENTRY_OVF      24

113 /* ERROR: invalid syntax.\nusage: roleadd ... */
114 #define M_ARUSAGE      25

116 /* ERROR: Invalid syntax.\nusage: roledel [-r] login\n" */
117 #define M_DRUSAGE      26

119 /* ERROR: Invalid syntax.\nusage: rolemod -u ... */
120 #define M_MRUSAGE      27

122 /* ERROR: project %s does not exist. Choose another. */
123 #define M_PROJ_NOTUSED      28
```

new/usr/src/cmd/oamuser/user/messages.h

3

```
125 /* WARNING: more than NPROJECTS_MAX(%d) projects specified. */
126 #define M_MAXPROJECTS 29

128 /* WARNING: Project entry exceeds 512 char: /etc/project entry truncated. */
129 #define M_PROJ_ENTRY_OVF 30

131 /* ERROR: Invalid key. */
132 #define M_INVALID_KEY 31

134 /* ERROR: Missing value specification. */
135 #define M_INVALID_VALUE 32

137 /* ERROR: Multiple definitions of key ``%s''. */
138 #define M_REDEFINED_KEY 33

140 /* ERROR: Roles must be modified with rolemod */
141 #define M_ISROLE 34

143 /* ERROR: Users must be modified with usermod */
144 #define M_ISUSER 35

146 /* WARNING: gid %d is reserved. */
147 #define M_RESERVED_GID 36

149 /* ERROR: Failed to read /etc/group file due to invalid entry or read error. */
150 #define M_READ_ERROR 37

152 /* ERROR: %s is too long. Choose another. */
153 #define M_TOO_LONG 38

155 #endif /* _MESSAGES_H */
```

new/usr/src/cmd/oamuser/user/useradd.c

1

```
*****
17433 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/user/useradd.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright 2008 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  */

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

32 #include <sys/types.h>
33 #include <sys/stat.h>
34 #include <sys/param.h>
35 #include <stdio.h>
36 #include <stdlib.h>
37 #include <ctype.h>
38 #include <limits.h>
39 #include <string.h>
40 #include <userdefs.h>
41 #include <errno.h>
42 #include <project.h>
43 #include <unistd.h>
44 #include <user_attr.h>
45 #include "users.h"
46 #include "messages.h"
47 #include "userdisp.h"
48 #include "funcs.h"

50 /*
51  * useradd [-u uid [-o] | -g group | -G group [[, group]...] | -d dir [-m]
52  *          | -s shell | -c comment | -k skel_dir | -b base_dir ]
53  *          [ -A authorization [, authorization ...]]
54  *          [ -P profile [, profile ...]]
55  *          [ -K key=value ]
56  *          [ -R role [, role ...]] [-p project [, project ...]] login
57  * useradd -D [ -g group | [ -b base_dir | -f inactive | -e expire |
58  *             -s shell | -k skel_dir ]
59  *          [ -A authorization [, authorization ...]]
60  *          [ -P profile [, profile ...]] [ -K key=value ]
```

new/usr/src/cmd/oamuser/user/useradd.c

2

```
61  *          [ -R role [, role ...]] [-p project [, project ...]] login
62  *
63  * This command adds new user logins to the system. Arguments are:
64  *
65  * uid - an integer
66  * group - an existing group's integer ID or char string name
67  * dir - home directory
68  * shell - a program to be used as a shell
69  * comment - any text string
70  * skel_dir - a skeleton directory
71  * base_dir - a directory
72  * login - a string of printable chars except colon(:)
73  * authorization - One or more comma separated authorizations defined
74  *                 in auth_attr(4).
75  * profile - One or more comma separated execution profiles defined
76  *          in prof_attr(4)
77  * role - One or more comma-separated role names defined in user_attr(4)
78  * project - One or more comma-separated project names or numbers
79  *
80  */

82 extern struct userdefs *getusrdef();
83 extern void dispusrdef();

85 static void cleanup();

87 extern uid_t findnextuid(void);
88 extern int check_perm(), valid_expire();
89 extern int putusrdef(), valid_uid();
90 extern int call_passmgmt(), edit_group(), create_home();
91 extern int edit_project();
92 extern int **valid_lgroup();
93 extern projid_t **valid_lproject();
94 extern void update_def(struct userdefs *);
95 extern void import_def(struct userdefs *);

97 static uid_t uid; /* new uid */
98 static char *logname; /* login name to add */
99 static struct userdefs *usrdefs; /* defaults for useradd */

101 char *cmdname;

103 static char homedir[ PATH_MAX + 1 ]; /* home directory */
104 static char gidstring[32]; /* group id string representation */
105 static gid_t gid; /* gid of new login */
106 static char uidstring[32]; /* user id string representation */
107 static char *uidstr = NULL; /* uid from command line */
108 static char *base_dir = NULL; /* base_dir from command line */
109 static char *group = NULL; /* group from command line */
110 static char *grps = NULL; /* multi groups from command line */
111 static char *dir = NULL; /* home dir from command line */
112 static char *shell = NULL; /* shell from command line */
113 static char *comment = NULL; /* comment from command line */
114 static char *skel_dir = NULL; /* skel dir from command line */
115 static long inact; /* inactive days */
116 static char *inactstr = NULL; /* inactive from command line */
117 static char inactstring[10]; /* inactivity string representation */
118 static char *expirestr = NULL; /* expiration date from command line */
119 static char *projects = NULL; /* project id's from command line */

121 static char *usertype = NULL; /* type of user, either role or normal */

123 typedef enum {
124     BASEDIR = 0,
125     SKELDIR,
126     SHELL
```

```

127 } path_opt_t;

130 static void valid_input(path_opt_t, const char *);

132 int
133 main(argc, argv)
134 int argc;
135 char *argv[];
136 {
137     int ch, ret, mflag = 0, oflag = 0, Dflag = 0, **gidlist;
138     projid_t **projlist;
139     char *ptr; /* loc in a str, may be set by strtol */
140     struct group *g_ptr;
141     struct project p_ptr;
142     char mybuf[PROJECT_BUFSZ];
143     struct stat statbuf; /* status buffer for stat */
144     int warning;
145     int busy = 0;
146     char **nargv; /* arguments for execvp of passmgmt */
147     int argindex; /* argument index into nargv */

149     cmdname = argv[0];

151     if (geteuid() != 0) {
152         errmsg(M_PERM_DENIED);
153         exit(EX_NO_PERM);
154     }

156     opterr = 0; /* no print errors from getopt */
157     usertype = getusertype(argv[0]);

159     change_key(USERATTR_TYPE_KW, usertype);

161     while ((ch = getopt(argc, argv,
162         "b:c:Dd:e:f:G:g:k:mop:s:u:A:P:R:K:")) != EOF)
163         switch (ch) {
164             case 'b':
165                 base_dir = optarg;
166                 break;

168             case 'c':
169                 comment = optarg;
170                 break;

172             case 'D':
173                 Dflag++;
174                 break;

176             case 'd':
177                 dir = optarg;
178                 break;

180             case 'e':
181                 expirestr = optarg;
182                 break;

184             case 'f':
185                 inactstr = optarg;
186                 break;

188             case 'G':
189                 grps = optarg;
190                 break;

192             case 'g':

```

```

193                 group = optarg;
194                 break;

196             case 'k':
197                 skel_dir = optarg;
198                 break;

200             case 'm':
201                 mflag++;
202                 break;

204             case 'o':
205                 oflag++;
206                 break;

208             case 'p':
209                 projects = optarg;
210                 break;

212             case 's':
213                 shell = optarg;
214                 break;

216             case 'u':
217                 uidstr = optarg;
218                 break;

220             case 'A':
221                 change_key(USERATTR_AUTHS_KW, optarg);
222                 break;

224             case 'P':
225                 change_key(USERATTR_PROFILES_KW, optarg);
226                 break;

228             case 'R':
229                 if (is_role(usertype)) {
230                     errmsg(M_ARUSAGE);
231                     exit(EX_SYNTAX);
232                 }
233                 change_key(USERATTR_ROLES_KW, optarg);
234                 break;

236             case 'K':
237                 change_key(NULL, optarg);
238                 break;

240             default:
241                 case '?':
242                     if (is_role(usertype))
243                         errmsg(M_ARUSAGE);
244                     else
245                         errmsg(M_AUSAGE);
246                     exit(EX_SYNTAX);
247                 }

249     /* get defaults for adding new users */
250     usrdefs = getusrdef(usertype);

252     if (Dflag) {
253         /* DISPLAY mode */

255         /* check syntax */
256         if (optind != argc) {
257             if (is_role(usertype))
258                 errmsg(M_ARUSAGE);

```

```

259         else
260             errmsg(M_AUSAGE);
261         exit(EX_SYNTAX);
262     }
263
264     if (uidstr != NULL || oflag || grps != NULL ||
265         dir != NULL || mflag || comment != NULL) {
266         if (is_role(usertype))
267             errmsg(M_ARUSAGE);
268         else
269             errmsg(M_AUSAGE);
270         exit(EX_SYNTAX);
271     }
272
273     /* Group must be an existing group */
274     if (group != NULL) {
275         switch (valid_group(group, &g_ptr, &warning)) {
276             case INVALID:
277                 errmsg(M_INVALID, group, "group id");
278                 exit(EX_BADARG);
279                 /*NOTREACHED*/
280             case TOOBIG:
281                 errmsg(M_TOOBIG, "gid", group);
282                 exit(EX_BADARG);
283                 /*NOTREACHED*/
284             case RESERVED:
285             case UNIQUE:
286                 errmsg(M_GRP_NOTUSED, group);
287                 exit(EX_NAME_NOT_EXIST);
288         }
289         if (warning)
290             warningmsg(warning, group);
291
292         usrdefs->defgroup = g_ptr->gr_gid;
293         usrdefs->defgname = g_ptr->gr_name;
294     }
295
296     /* project must be an existing project */
297     if (projects != NULL) {
298         switch (valid_project(projects, &p_ptr, mybuf,
299                             sizeof(mybuf), &warning)) {
300             case INVALID:
301                 errmsg(M_INVALID, projects, "project id");
302                 exit(EX_BADARG);
303                 /*NOTREACHED*/
304             case TOOBIG:
305                 errmsg(M_TOOBIG, "projid", projects);
306                 exit(EX_BADARG);
307                 /*NOTREACHED*/
308             case UNIQUE:
309                 errmsg(M_PROJ_NOTUSED, projects);
310                 exit(EX_NAME_NOT_EXIST);
311         }
312         if (warning)
313             warningmsg(warning, projects);
314
315         usrdefs->defproj = p_ptr.pj_projid;
316         usrdefs->defprojname = p_ptr.pj_name;
317     }
318
319     /* base_dir must be an existing directory */
320     if (base_dir != NULL) {
321         valid_input(BASEDIR, base_dir);
322         usrdefs->defparent = base_dir;
323     }
324

```

```

326         /* inactivity period is an integer */
327         if (inactstr != NULL) {
328             /* convert inactstr to integer */
329             inact = strtol(inactstr, &ptr, 10);
330             if (*ptr || inact < 0) {
331                 errmsg(M_INVALID, inactstr,
332                     "inactivity period");
333                 exit(EX_BADARG);
334             }
335
336             usrdefs->definact = inact;
337         }
338
339         /* expiration string is a date, newer than today */
340         if (expirestr != NULL) {
341             if (*expirestr) {
342                 if (valid_expire(expirestr, (time_t *)0)
343                     == INVALID) {
344                     errmsg(M_INVALID, expirestr,
345                         "expiration date");
346                     exit(EX_BADARG);
347                 }
348                 usrdefs->defexpire = expirestr;
349             } else
350                 /* Unset the expiration date */
351                 usrdefs->defexpire = "";
352         }
353
354         if (shell != NULL) {
355             valid_input(SHELL, shell);
356             usrdefs->defshell = shell;
357         }
358         if (skel_dir != NULL) {
359             valid_input(SKELDIR, skel_dir);
360             usrdefs->defskel = skel_dir;
361         }
362         update_def(usrdefs);
363
364         /* change defaults for useradd */
365         if (putusrdef(usrdefs, usertype) < 0) {
366             errmsg(M_UPDATE, "created");
367             exit(EX_UPDATE);
368         }
369
370         /* Now, display */
371         dispusrdef(stdout, (D_ALL & ~D_RID), usertype);
372         exit(EX_SUCCESS);
373     }
374 }
375
376 /* ADD mode */
377
378 /* check syntax */
379 if (optind != argc - 1 || (skel_dir != NULL && !mflag)) {
380     if (is_role(usertype))
381         errmsg(M_ARUSAGE);
382     else
383         errmsg(M_AUSAGE);
384     exit(EX_SYNTAX);
385 }
386
387 logname = argv[optind];
388 switch (valid_login(logname, (struct passwd **)NULL, &warning)) {
389     case INVALID:
390         errmsg(M_INVALID, logname, "login name");

```



```

391         exit(EX_BADARG);
392         /*NOTREACHED*/

394     case NOTUNIQUE:
395         errmsg(M_USED, logname);
396         exit(EX_NAME_EXISTS);
397         /*NOTREACHED*/

399     case LONGNAME:
400         errmsg(M_TOO_LONG, logname);
401         exit(EX_BADARG);
402         /*NOTREACHED*/
403     }

405     if (warning)
406         warningmsg(warning, logname);
407     if (uidstr != NULL) {
408         /* convert uidstr to integer */
409         errno = 0;
410         uid = (uid_t)strtol(uidstr, &ptr, (int)10);
411         if (*ptr || errno == ERANGE) {
412             errmsg(M_INVALID, uidstr, "user id");
413             exit(EX_BADARG);
414         }

416         switch (valid_uid(uid, NULL)) {
417             case NOTUNIQUE:
418                 if (!oflag) {
419                     /* override not specified */
420                     errmsg(M_UID_USED, uid);
421                     exit(EX_ID_EXISTS);
422                 }
423                 break;
424             case RESERVED:
425                 errmsg(M_RESERVED, uid);
426                 break;
427             case TOOBIG:
428                 errmsg(M_TOOBIG, "uid", uid);
429                 exit(EX_BADARG);
430                 break;
431         }

433     } else {

435         if ((uid = findnextuid()) < 0) {
436             errmsg(M_INVALID, "default id", "user id");
437             exit(EX_ID_EXISTS);
438         }
439     }

441     if (group != NULL) {
442         switch (valid_group(group, &g_ptr, &warning)) {
443             case INVALID:
444                 errmsg(M_INVALID, group, "group id");
445                 exit(EX_BADARG);
446                 /*NOTREACHED*/
447             case TOOBIG:
448                 errmsg(M_TOOBIG, "gid", group);
449                 exit(EX_BADARG);
450                 /*NOTREACHED*/
451             case RESERVED:
452             case UNIQUE:
453                 errmsg(M_GRP_NOTUSED, group);
454                 exit(EX_NAME_NOT_EXIST);
455                 /*NOTREACHED*/
456         }

```

```

458         if (warning)
459             warningmsg(warning, group);
460         gid = g_ptr->gr_gid;

462     } else gid = usrdefs->defgroup;

464     if (grps != NULL) {
465         if (!*grps)
466             /* ignore -G "" */
467             grps = (char *)0;
468         else if (!(gidlist = valid_lgroup(grps, gid)))
469             exit(EX_BADARG);
470     }

472     if (projects != NULL) {
473         if (!*projects)
474             projects = (char *)0;
475         else if (!(projlist = valid_lproject(projects)))
476             exit(EX_BADARG);
477     }

479     /* if base_dir is provided, check its validity; otherwise default */
480     if (base_dir != NULL)
481         valid_input(BASEDIR, base_dir);
482     else
483         base_dir = usrdefs->defparent;

485     if (dir == NULL) {
486         /* set homedir to home directory made from base_dir */
487         (void) sprintf(homedir, "%s/%s", base_dir, logname);

489     } else if (REL_PATH(dir)) {
490         errmsg(M_RELPATH, dir);
491         exit(EX_BADARG);

493     } else
494         (void) strcpy(homedir, dir);

496     if (mflag) {
497         /* Does home dir. already exist? */
498         if (stat(homedir, &statbuf) == 0) {
499             /* directory exists - don't try to create */
500             mflag = 0;

502                 if (check_perm(statbuf, uid, gid, S_IXOTH) != 0)
503                     errmsg(M_NO_PERM, logname, homedir);
504             }
505         }
506     /*
507     * if shell, skel_dir are provided, check their validity.
508     * Otherwise default.
509     */
510     if (shell != NULL)
511         valid_input(SHELL, shell);
512     else
513         shell = usrdefs->defshell;

515     if (skel_dir != NULL)
516         valid_input(SKELDIR, skel_dir);
517     else
518         skel_dir = usrdefs->defskel;

520     if (inactstr != NULL) {
521         /* convert inactstr to integer */
522         inact = strtol(inactstr, &ptr, 10);

```

```

523         if (*ptr || inact < 0) {
524             errmsg(M_INVALID, inactstr, "inactivity period");
525             exit(EX_BADARG);
526         }
527     } else inact = usrdefs->definact;

529     /* expiration string is a date, newer than today */
530     if (expirestr != NULL) {
531         if (*expirestr) {
532             if (valid_expire(expirestr, (time_t *)0) == INVALID) {
533                 errmsg(M_INVALID, expirestr, "expiration date");
534                 exit(EX_BADARG);
535             }
536             usrdefs->defexpire = expirestr;
537         } else
538             /* Unset the expiration date */
539             expirestr = (char *)0;

541     } else expirestr = usrdefs->defexpire;

543     import_def(usrdefs);

545     /* must now call passmgmt */

547     /* set up arguments to passmgmt in nargv array */
548     nargv = malloc((30 + nkeys * 2) * sizeof(char *));
549     argindex = 0;
550     nargv[argindex++] = PASSMGMT;
551     nargv[argindex++] = "-a";        /* add */

553     if (comment != NULL) {
554         /* comment */
555         nargv[argindex++] = "-c";
556         nargv[argindex++] = comment;
557     }

559     /* flags for home directory */
560     nargv[argindex++] = "-h";
561     nargv[argindex++] = homedir;

563     /* set gid flag */
564     nargv[argindex++] = "-g";
565     (void) sprintf(gidstring, "%u", gid);
566     nargv[argindex++] = gidstring;

568     /* shell */
569     nargv[argindex++] = "-s";
570     nargv[argindex++] = shell;

572     /* set inactive */
573     nargv[argindex++] = "-f";
574     (void) sprintf(inactstring, "%ld", inact);
575     nargv[argindex++] = inactstring;

577     /* set expiration date */
578     if (expirestr != NULL) {
579         nargv[argindex++] = "-e";
580         nargv[argindex++] = expirestr;
581     }

583     /* set uid flag */
584     nargv[argindex++] = "-u";
585     (void) sprintf(uidstring, "%u", uid);
586     nargv[argindex++] = uidstring;

588     if (oflag) nargv[argindex++] = "-o";

```

```

590     if (nkeys > 1)
591         addkey_args(nargv, &argindex);

593     /* finally - login name */
594     nargv[argindex++] = logname;

596     /* set the last to null */
597     nargv[argindex++] = NULL;

599     /* now call passmgmt */
600     ret = PEX_FAILED;
601     /*
602     * If call_passmgmt fails for any reason other than PEX_BADUID, exit
603     * is invoked with an appropriate error message. If PEX_BADUID is
604     * returned, then if the user specified the ID, exit is invoked
605     * with an appropriate error message. Otherwise we try to pick a
606     * different ID and try again. If we run out of IDs, i.e. no more
607     * users can be created, then -l is returned and we terminate via exit.
608     * If PEX_BUSY is returned we increment a count, since we will stop
609     * trying if PEX_BUSY reaches 3. For PEX_SUCCESS we immediately
610     * terminate the loop.
611     */
612     while (busy < 3 && ret != PEX_SUCCESS) {
613         switch (ret = call_passmgmt(nargv)) {
614             case PEX_SUCCESS:
615                 break;
616             case PEX_BUSY:
617                 busy++;
618                 break;
619             case PEX_HOSED_FILES:
620                 errmsg(M_HOSED_FILES);
621                 exit(EX_INCONSISTENT);
622                 break;

624             case PEX_SYNTAX:
625             case PEX_BADARG:
626                 /* should NEVER occur that passmgmt usage is wrong */
627                 if (is_role(usertype))
628                     errmsg(M_ARUSAGE);
629                 else
630                     errmsg(M_AUSAGE);
631                 exit(EX_SYNTAX);
632                 break;

634             case PEX_BADUID:
635                 /*
636                 * The uid has been taken. If it was specified by a
637                 * user, then we must fail. Otherwise, keep trying
638                 * to get a good uid until we run out of IDs.
639                 */
640                 if (uidstr != NULL) {
641                     errmsg(M_UID_USED, uid);
642                     exit(EX_ID_EXISTS);
643                 } else {
644                     if ((uid = findnextuid()) < 0) {
645                         errmsg(M_INVALID, "default id",
646                             "user id");
647                         exit(EX_ID_EXISTS);
648                     }
649                     (void) sprintf(uidstring, "%u", uid);
650                 }
651                 break;

653             case PEX_BADNAME:
654                 /* invalid loname */

```

```
655         errmsg(M_USED, logname);
656         exit(EX_NAME_EXISTS);
657         break;

659     default:
660         errmsg(M_UPDATE, "created");
661         exit(ret);
662         break;
663     }
664 }
665 if (busy == 3) {
666     errmsg(M_UPDATE, "created");
667     exit(ret);
668 }

670 /* add group entry */
671 if ((grps != NULL) && edit_group(logname, (char *)0, gidlist, 0)) {
672     errmsg(M_UPDATE, "created");
673     cleanup(logname);
674     exit(EX_UPDATE);
675 }

677 /* update project database */
678 if ((projects != NULL) &&
679     edit_project(logname, (char *)NULL, projlist, 0)) {
680     errmsg(M_UPDATE, "created");
681     cleanup(logname);
682     exit(EX_UPDATE);
683 }

685 /* create home directory */
686 if (mflag &&
687     (create_home(homedir, skel_dir, uid, gid) != EX_SUCCESS)) {
688     (void) edit_group(logname, (char *)0, (int **)0, 1);
689     cleanup(logname);
690     exit(EX_HOMEDIR);
691 }

693     return (ret);
694 }
unchanged portion omitted
```

new/usr/src/cmd/oamuser/user/useradd.dfl

1

\*\*\*\*\*

1242 Sat Jun 1 10:02:48 2013

new/usr/src/cmd/oamuser/user/useradd.dfl

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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

24 # The EXCEED_TRAD indicates the action when the traditional login name
25 # length limit of eight characters is exceeded. The value "warning"
26 # means to issue a warning message and continue. This is the default.
27 # The value "error" means to issue an error message and terminate.
28 # The value "silent" means to continue without issuing any message.
29 #
30 EXCEED_TRAD=warning
31 #EXCEED_TRAD=error
32 #EXCEED_TRAD=silent
```

new/usr/src/cmd/oamuser/user/usermod.c

1

```
*****
15671 Sat Jun 1 10:02:48 2013
new/usr/src/cmd/oamuser/user/usermod.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
24 * Copyright 2008 Sun Microsystems, Inc. All rights reserved.
25 * Use is subject to license terms.
26 */

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

33 #include <sys/types.h>
34 #include <sys/stat.h>
35 #include <sys/param.h>
36 #include <stdio.h>
37 #include <stdlib.h>
38 #include <ctype.h>
39 #include <limits.h>
40 #include <string.h>
41 #include <userdefs.h>
42 #include <user_attr.h>
43 #include <nss_dbdefs.h>
44 #include <errno.h>
45 #include <project.h>
46 #include "users.h"
47 #include "messages.h"
48 #include "funcs.h"

50 /*
51 * usermod [-u uid [-o] | -g group | -G group [[,group]...] | -d dir [-m]
52 *          | -s shell | -c comment | -l new_logname]
53 *          [-f inactive | -e expire]
54 *          [-A authorization [, authorization ...]]
55 *          [-P profile [, profile ...]]
56 *          [-R role [, role ...]]
57 *          [-K key=value]
58 *          [-p project [, project]] login
59 *
60 *      This command adds new user logins to the system.  Arguments are:
```

new/usr/src/cmd/oamuser/user/usermod.c

2

```
61 *
62 *      uid - an integer less than MAXUID
63 *      group - an existing group's integer ID or char string name
64 *      dir - a directory
65 *      shell - a program to be used as a shell
66 *      comment - any text string
67 *      skel_dir - a directory
68 *      base_dir - a directory
69 *      rid - an integer less than 2**16 (USHORT)
70 *      login - a string of printable chars except colon (:)
71 *      inactive - number of days a login maybe inactive before it is locked
72 *      expire - date when a login is no longer valid
73 *      authorization - One or more comma separated authorizations defined
74 *                    in auth_attr(4).
75 *      profile - One or more comma separated execution profiles defined
76 *              in prof_attr(4)
77 *      role - One or more comma-separated role names defined in user_attr(4)
78 *      key=value - One or more -K options each specifying a valid user_attr(4)
79 *                attribute.
80 *
81 */

83 extern int **valid_lgroup(), isbusy();
84 extern int valid_uid(), check_perm(), create_home(), move_dir();
85 extern int valid_expire(), edit_group(), call_passmgmt();
86 extern projid_t **valid_lproject();

88 static uid_t uid;          /* new uid */
89 static gid_t gid;         /* gid of new login */
90 static char *new_logname = NULL; /* new login name with -l option */
91 static char *uidstr = NULL; /* uid from command line */
92 static char *group = NULL; /* group from command line */
93 static char *grps = NULL; /* multi groups from command line */
94 static char *dir = NULL; /* home dir from command line */
95 static char *shell = NULL; /* shell from command line */
96 static char *comment = NULL; /* comment from command line */
97 static char *loginame = NULL; /* login name to add */
98 static char *inactstr = NULL; /* inactive from command line */
99 static char *expire = NULL; /* expiration date from command line */
100 static char *projects = NULL; /* project ids from command line */
101 static char *usertype;

103 char *cmdname;
104 static char gidstring[32], uidstring[32];
105 char inactstring[10];

107 char *
108 strcpmalloc(str)
109 char *str;
110 {
111     if (str == NULL)
112         return (NULL);
113
114     return (strdup(str));
115 }

unchanged_portion_omitted

141 int
142 main(argc, argv)
143 int argc;
144 char **argv;
145 {
146     int ch, ret = EX_SUCCESS, call_pass = 0, oflag = 0;
147     int tries, mflag = 0, inact, **gidlist, flag = 0;
148     boolean_t fail_if_busy = B_FALSE;
149     char *ptr;
```

```

150     struct passwd *pstruct;          /* password struct for login */
151     struct passwd *pw;
152     struct group *g_ptr;             /* validated group from -g */
153     struct stat statbuf;             /* status buffer for stat */
154 #ifndef att
155     FILE *pwf;                       /* fille ptr for opened passwd file */
156 #endif
157     int warning;
158     projid_t **projlist;
159     char **nargv;                    /* arguments for execvp of passgmt */
160     int argindex;                    /* argument index into nargv */
161     userattr_t *ua;
162     char *val;
163     int isrole;                       /* current account is role */

165     cmdname = argv[0];

167     if (geteuid() != 0) {
168         errmsg(M_PERM_DENIED);
169         exit(EX_NO_PERM);
170     }

172     opterr = 0;                       /* no print errors from getopt */
173     /* get user type based on the program name */
174     usertype = getusertype(argv[0]);

176     while ((ch = getopt(argc, argv,
177         "c:d:e:f:G:g:l:mop:s:u:A:P:R:K:")) != EOF)
178         switch (ch) {
179             case 'c':
180                 comment = optarg;
181                 flag++;
182                 break;
183             case 'd':
184                 dir = optarg;
185                 fail_if_busy = B_TRUE;
186                 flag++;
187                 break;
188             case 'e':
189                 expire = optarg;
190                 flag++;
191                 break;
192             case 'f':
193                 inactstr = optarg;
194                 flag++;
195                 break;
196             case 'G':
197                 grps = optarg;
198                 flag++;
199                 break;
200             case 'g':
201                 group = optarg;
202                 fail_if_busy = B_TRUE;
203                 flag++;
204                 break;
205             case 'l':
206                 new_logname = optarg;
207                 fail_if_busy = B_TRUE;
208                 flag++;
209                 break;
210             case 'm':
211                 mflag++;
212                 flag++;
213                 fail_if_busy = B_TRUE;
214                 break;
215             case 'o':

```

```

216                 oflag++;
217                 flag++;
218                 fail_if_busy = B_TRUE;
219                 break;
220             case 'p':
221                 projects = optarg;
222                 flag++;
223                 break;
224             case 's':
225                 shell = optarg;
226                 flag++;
227                 break;
228             case 'u':
229                 uidstr = optarg;
230                 flag++;
231                 fail_if_busy = B_TRUE;
232                 break;
233             case 'A':
234                 change_key(USERATTR_AUTHS_KW, optarg);
235                 flag++;
236                 break;
237             case 'P':
238                 change_key(USERATTR_PROFILES_KW, optarg);
239                 flag++;
240                 break;
241             case 'R':
242                 change_key(USERATTR_ROLES_KW, optarg);
243                 flag++;
244                 break;
245             case 'K':
246                 change_key(NULL, optarg);
247                 flag++;
248                 break;
249             default:
250                 case '?':
251                     if (is_role(usertype))
252                         errmsg(M_MRUSAGE);
253                     else
254                         errmsg(M_MUSAGE);
255                     exit(EX_SYNTAX);
256                 }

258     if (optind != argc - 1 || flag == 0) {
259         if (is_role(usertype))
260             errmsg(M_MRUSAGE);
261         else
262             errmsg(M_MUSAGE);
263         exit(EX_SYNTAX);
264     }

266     if ((!uidstr && oflag) || (mflag && !dir)) {
267         if (is_role(usertype))
268             errmsg(M_MRUSAGE);
269         else
270             errmsg(M_MUSAGE);
271         exit(EX_SYNTAX);
272     }

274     logname = argv[optind];

276     /* Determine whether the account is a role or not */
277     if ((ua = getusernam(logname)) == NULL ||
278         (val = kva_match(ua->attr, USERATTR_TYPE_KW)) == NULL ||
279         strcmp(val, USERATTR_TYPE_NONADMIN_KW) != 0)
280         isrole = 0;
281     else

```

```

282         isrole = 1;

284     /* Verify that rolemod is used for roles and usermod for users */
285     if (isrole != is_role(usertype)) {
286         if (isrole)
287             errormsg(M_ISROLE);
288         else
289             errormsg(M_ISUSER);
290         exit(EX_SYNTAX);
291     }

293     /* Set the usertype key; defaults to the commandline */
294     usertype = getsetdefval(USERATTR_TYPE_KW, usertype);

296     if (is_role(usertype)) {
297         /* Roles can't have roles */
298         if (getsetdefval(USERATTR_ROLES_KW, NULL) != NULL) {
299             errormsg(M_MRUSAGE);
300             exit(EX_SYNTAX);
301         }
302         /* If it was an ordinary user, delete its roles */
303         if (!isrole)
304             change_key(USERATTR_ROLES_KW, "");
305     }

307 #ifdef att
308     pw = getpwnam(logname);
309 #else
310     /*
311     * Do this with fgetpwent to make sure we are only looking on local
312     * system (since passgmt only works on local system).
313     */
314     if ((pwf = fopen("/etc/passwd", "r")) == NULL) {
315         errormsg(M_OOPS, "open", "/etc/passwd");
316         exit(EX_FAILURE);
317     }
318     while ((pw = fgetpwent(pwf)) != NULL)
319         if (strcmp(pw->pw_name, logname) == 0)
320             break;

322     fclose(pwf);
323 #endif

325     if (pw == NULL) {
326         char          pwdb[NSS_BUFLLEN_PASSWD];
327         struct passwd  pwd;

329         if (getpwnam_r(logname, &pwd, pwdb, sizeof (pwdb)) == NULL) {
330             /* This user does not exist. */
331             errormsg(M_EXIST, logname);
332             exit(EX_NAME_NOT_EXIST);
333         } else {
334             /* This user exists in non-local name service. */
335             errormsg(M_NONLOCAL, logname);
336             exit(EX_NOT_LOCAL);
337         }
338     }

340     pstruct = passwd_cpmalloc(pw);

342     /*
343     * We can't modify a logged in user if any of the following
344     * are being changed:
345     * uid (-u & -o), group (-g), home dir (-m), loginname (-l).
346     * If none of those are specified it is okay to go ahead
347     * some types of changes only take effect on next login, some

```

```

348     * like authorisations and profiles take effect instantly.
349     * One might think that -K type=role should require that the
350     * user not be logged in, however this would make it very
351     * difficult to make the root account a role using this command.
352     */
353     if (isbusy(logname)) {
354         if (fail_if_busy) {
355             errormsg(M_BUSY, logname, "change");
356             exit(EX_BUSY);
357         }
358         warningmsg(WARN_LOGGED_IN, logname);
359     }

361     if (new_logname && strcmp(new_logname, logname)) {
362         switch (valid_login(new_logname, (struct passwd **)NULL,
363             &warning)) {
364             case INVALID:
365                 errormsg(M_INVALID, new_logname, "login name");
366                 exit(EX_BADARG);
367                 /*NOTREACHED*/

369             case NOTUNIQUE:
370                 errormsg(M_USED, new_logname);
371                 exit(EX_NAME_EXISTS);
372                 /*NOTREACHED*/

374             case LONGNAME:
375                 errormsg(M_TOO_LONG, new_logname);
376                 exit(EX_BADARG);
377                 /*NOTREACHED*/

379             default:
380                 call_pass = 1;
381                 break;
382         }
383         if (warning)
384             warningmsg(warning, logname);
385     }

387     if (uidstr) {
388         /* convert uidstr to integer */
389         errno = 0;
390         uid = (uid_t)strtol(uidstr, &ptr, (int)10);
391         if (*ptr || errno == ERANGE) {
392             errormsg(M_INVALID, uidstr, "user id");
393             exit(EX_BADARG);
394         }

396         if (uid != pstruct->pw_uid) {
397             switch (valid_uid(uid, NULL)) {
398                 case NOTUNIQUE:
399                     if (!oflag) {
400                         /* override not specified */
401                         errormsg(M_UID_USED, uid);
402                         exit(EX_ID_EXISTS);
403                     }
404                     break;
405                 case RESERVED:
406                     errormsg(M_RESERVED, uid);
407                     break;
408                 case TOOBIG:
409                     errormsg(M_TOOBIG, "uid", uid);
410                     exit(EX_BADARG);
411                     break;
412             }

```

```

414         call_pass = 1;
416     } else {
417         /* uid's the same, so don't change anything */
418         uidstr = NULL;
419         oflag = 0;
420     }
422 } else uid = pstruct->pw_uid;
424 if (group) {
425     switch (valid_group(group, &g_ptr, &warning)) {
426     case INVALID:
427         errmsg(M_INVALID, group, "group id");
428         exit(EX_BADARG);
429         /*NOTREACHED*/
430     case TOOBIG:
431         errmsg(M_TOOBIG, "gid", group);
432         exit(EX_BADARG);
433         /*NOTREACHED*/
434     case UNIQUE:
435         errmsg(M_GRP_NOTUSED, group);
436         exit(EX_NAME_NOT_EXIST);
437         /*NOTREACHED*/
438     case RESERVED:
439         gid = (gid_t)strtol(group, &ptr, (int)10);
440         errmsg(M_RESERVED_GID, gid);
441         break;
442     }
443     if (warning)
444         warningmsg(warning, group);
446     if (g_ptr != NULL)
447         gid = g_ptr->gr_gid;
448     else
449         gid = pstruct->pw_gid;
451     /* call passmgmt if gid is different, else ignore group */
452     if (gid != pstruct->pw_gid)
453         call_pass = 1;
454     else group = NULL;
456 } else gid = pstruct->pw_gid;
458 if (grps && *grps) {
459     if (!(gidlist = valid_lgroup(grps, gid)))
460         exit(EX_BADARG);
461 } else
462     gidlist = (int **)0;
464 if (projects && *projects) {
465     if (!(projlist = valid_lproject(projects)))
466         exit(EX_BADARG);
467 } else
468     projlist = (projid_t **)0;
470 if (dir) {
471     if (REL_PATH(dir)) {
472         errmsg(M_RELPATH, dir);
473         exit(EX_BADARG);
474     }
475     if (strcmp(pstruct->pw_dir, dir) == 0) {
476         /* home directory is the same so ignore dflag & mflag */
477         dir = NULL;
478         mflag = 0;
479     } else call_pass = 1;

```

```

480     }
482     if (mflag) {
483         if (stat(dir, &statbuf) == 0) {
484             /* Home directory exists */
485             if (check_perm(statbuf, pstruct->pw_uid,
486                 pstruct->pw_gid, S_IWOTH|S_IXOTH) != 0) {
487                 errmsg(M_NO_PERM, logname, dir);
488                 exit(EX_NO_PERM);
489             }
491         } else ret = create_home(dir, NULL, uid, gid);
493         if (ret == EX_SUCCESS)
494             ret = move_dir(pstruct->pw_dir, dir, logname);
496         if (ret != EX_SUCCESS)
497             exit(ret);
498     }
500     if (shell) {
501         if (REL_PATH(shell)) {
502             errmsg(M_RELPATH, shell);
503             exit(EX_BADARG);
504         }
505         if (strcmp(pstruct->pw_shell, shell) == 0) {
506             /* ignore s option if shell is not different */
507             shell = NULL;
508         } else {
509             if (stat(shell, &statbuf) < 0 ||
510                 (statbuf.st_mode & S_IFMT) != S_IFREG ||
511                 (statbuf.st_mode & 0555) != 0555) {
513                 errmsg(M_INVALID, shell, "shell");
514                 exit(EX_BADARG);
515             }
517             call_pass = 1;
518         }
519     }
521     if (comment)
522         /* ignore comment if comment is not changed */
523         if (strcmp(pstruct->pw_comment, comment))
524             call_pass = 1;
525     else
526         comment = NULL;
528     /* inactive string is a positive integer */
529     if (inactstr) {
530         /* convert inactstr to integer */
531         inact = (int)strtol(inactstr, &ptr, 10);
532         if (*ptr || inact < 0) {
533             errmsg(M_INVALID, inactstr, "inactivity period");
534             exit(EX_BADARG);
535         }
536         call_pass = 1;
537     }
539     /* expiration string is a date, newer than today */
540     if (expire) {
541         if (*expire &&
542             valid_expire(expire, (time_t *)0) == INVALID) {
543             errmsg(M_INVALID, expire, "expiration date");
544             exit(EX_BADARG);
545         }

```



```

546         call_pass = 1;
547     }

549     if (nkeys > 0)
550         call_pass = 1;

552     /* that's it for validations - now do the work */

554     if (grps) {
555         /* redefine login's supplementary group memberships */
556         ret = edit_group(logname, new_logname, gidlist, 1);
557         if (ret != EX_SUCCESS) {
558             errmsg(M_UPDATE, "modified");
559             exit(ret);
560         }
561     }
562     if (projects) {
563         ret = edit_project(logname, (char *)NULL, projlist, 0);
564         if (ret != EX_SUCCESS) {
565             errmsg(M_UPDATE, "modified");
566             exit(ret);
567         }
568     }

571     if (!call_pass) exit(ret);

573     /* only get to here if need to call passmgmt */
574     /* set up arguments to passmgmt in nargv array */
575     nargv = malloc((30 + nkeys * 2) * sizeof(char *));

577     argindex = 0;
578     nargv[argindex++] = PASSMGMT;
579     nargv[argindex++] = "-m";      /* modify */

581     if (comment) { /* comment */
582         nargv[argindex++] = "-c";
583         nargv[argindex++] = comment;
584     }

586     if (dir) {
587         /* flags for home directory */
588         nargv[argindex++] = "-h";
589         nargv[argindex++] = dir;
590     }

592     if (group) {
593         /* set gid flag */
594         nargv[argindex++] = "-g";
595         (void) sprintf(gidstring, "%u", gid);
596         nargv[argindex++] = gidstring;
597     }

599     if (shell) { /* shell */
600         nargv[argindex++] = "-s";
601         nargv[argindex++] = shell;
602     }

604     if (inactstr) {
605         nargv[argindex++] = "-f";
606         nargv[argindex++] = inactstr;
607     }

609     if (expire) {
610         nargv[argindex++] = "-e";
611         nargv[argindex++] = expire;

```

```

612     }

614     if (uidstr) { /* set uid flag */
615         nargv[argindex++] = "-u";
616         (void) sprintf(uidstring, "%u", uid);
617         nargv[argindex++] = uidstring;
618     }

620     if (oflag) nargv[argindex++] = "-o";

622     if (new_logname) { /* redefine login name */
623         nargv[argindex++] = "-l";
624         nargv[argindex++] = new_logname;
625     }

627     if (nkeys > 0)
628         addkey_args(nargv, &argindex);

630     /* finally - login name */
631     nargv[argindex++] = logname;

633     /* set the last to null */
634     nargv[argindex++] = NULL;

636     /* now call passmgmt */
637     ret = PEX_FAILED;
638     for (tries = 3; ret != PEX_SUCCESS && tries--;) {
639         switch (ret = call_passmgmt(nargv)) {
640             case PEX_SUCCESS:
641                 break;
642             case PEX_HOSED_FILES:
643                 errmsg(M_HOSED_FILES);
644                 exit(EX_INCONSISTENT);
645                 break;
646             case PEX_SYNTAX:
647                 break;
648             case PEX_BADARG:
649                 /* should NEVER occur that passmgmt usage is wrong */
650                 if (is_role(usertype))
651                     errmsg(M_MRUSAGE);
652                 else
653                     errmsg(M_MUSAGE);
654                 exit(EX_SYNTAX);
655                 break;
656             case PEX_BADUID:
657                 /* uid in use - shouldn't happen print message anyway */
658                 errmsg(M_UID_USED, uid);
659                 exit(EX_ID_EXISTS);
660                 break;
661             case PEX_BADNAME:
662                 /* invalid loname */
663                 errmsg(M_USED, logname);
664                 exit(EX_NAME_EXISTS);
665                 break;
666             default:
667                 errmsg(M_UPDATE, "modified");
668                 exit(ret);
669                 break;
670         }
671     }
672     if (tries == 0) {

```

new/usr/src/cmd/oamuser/user/usermod.c

11

```
678         errmsg(M_UPDATE, "modified");
679     }
681     exit(ret);
682     /*NOTREACHED*/
683 }
unchanged_portion_omitted
```

```

*****
44951 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/prstat/prstat.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
25  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
26  * Use is subject to license terms.
27  *
28  * Portions Copyright 2009 Chad Mynhier
29  */

31 #include <sys/types.h>
32 #include <sys/resource.h>
33 #include <sys/loadavg.h>
34 #include <sys/time.h>
35 #include <sys/pset.h>
36 #include <sys/vm_usage.h>
37 #include <zone.h>
38 #include <libzonecfg.h>

40 #include <stdio.h>
41 #include <stdlib.h>
42 #include <unistd.h>
43 #include <dirent.h>
44 #include <string.h>
45 #include <errno.h>
46 #include <poll.h>
47 #include <ctype.h>
48 #include <fcntl.h>
49 #include <limits.h>
50 #include <signal.h>
51 #include <time.h>
52 #include <project.h>

54 #include <langinfo.h>
55 #include <libintl.h>
56 #include <locale.h>

58 #include "prstat.h"
59 #include "prutil.h"
60 #include "prtable.h"

```

```

61 #include "prsort.h"
62 #include "prfile.h"

64 /*
65  * x86 <sys/regs.h> ERR conflicts with <curses.h> ERR. For the purposes
66  * of this file, we care about the curses.h ERR so include that last.
67  */

69 #if defined(ERR)
70 #undef ERR
71 #endif

73 #ifndef TEXT_DOMAIN
74 #define TEXT_DOMAIN "SYS_TEST" /* should be defined by cc -D */
75 #endif

77 #include <curses.h>
78 #include <term.h>

80 #define LOGIN_WIDTH 8
81 #define ZONE_WIDTH 28
82 #define PROJECT_WIDTH 28

84 #define PSINFO_HEADER_PROC \
85 " PID USERNAME SIZE RSS STATE PRI NICE TIME CPU PROCESS/NLWP "
86 #define PSINFO_HEADER_PROC_LGRP \
87 " PID USERNAME SIZE RSS STATE PRI NICE TIME CPU LGRP PROCESS/NLWP "
88 #define PSINFO_HEADER_LWP \
89 " PID USERNAME SIZE RSS STATE PRI NICE TIME CPU PROCESS/LWPID "
90 #define PSINFO_HEADER_LWP_LGRP \
91 " PID USERNAME SIZE RSS STATE PRI NICE TIME CPU LGRP PROCESS/LWPID "
92 #define USAGE_HEADER_PROC \
93 " PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/NLWP "
94 #define USAGE_HEADER_LWP \
95 " PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWPID "
96 #define USER_HEADER_PROC \
97 " NPROC USERNAME SWAP RSS MEMORY TIME CPU "
98 #define USER_HEADER_LWP \
99 " NLWP USERNAME SWAP RSS MEMORY TIME CPU "
100 #define TASK_HEADER_PROC \
101 "TASKID NPROC SWAP RSS MEMORY TIME CPU PROJECT "
102 #define TASK_HEADER_LWP \
103 "TASKID NLWP SWAP RSS MEMORY TIME CPU PROJECT "
104 #define PROJECT_HEADER_PROC \
105 "PROJID NPROC SWAP RSS MEMORY TIME CPU PROJECT "
106 #define PROJECT_HEADER_LWP \
107 "PROJID NLWP SWAP RSS MEMORY TIME CPU PROJECT "
108 #define ZONE_HEADER_PROC \
109 "ZONEID NPROC SWAP RSS MEMORY TIME CPU ZONE "
110 #define ZONE_HEADER_LWP \
111 "ZONEID NLWP SWAP RSS MEMORY TIME CPU ZONE "
112 #define PSINFO_LINE \
113 "%6d %-8s %5s %5s %-6s %3s %3s %9s %3.3s%% %-.16s/%d"
114 #define PSINFO_LINE_LGRP \
115 "%6d %-8s %5s %5s %-6s %3s %3s %9s %3.3s%% %4d %-.16s/%d"
116 #define USAGE_LINE \
117 "%6d %-8s %3.3s %3.3s %3.3s %3.3s %3.3s %3.3s %3.3s %3.3s \% "
118 "%3.3s %3.3s %-.12s/%d"
119 #define USER_LINE \
120 "%6d %-8s %5.5s %5.5s %3.3s%% %9s %3.3s%%"
121 #define TASK_LINE \
122 "%6d %8d %5s %5s %3.3s%% %9s %3.3s%% %28s"
123 #define PROJECT_LINE \
124 "%6d %8d %5s %5s %3.3s%% %9s %3.3s%% %28s"

```

```

125 #define ZONE_LINE \
126 "%6d %8d %5s %5s %3.3s%% %9s %3.3s%% %28s"

128 #define TOTAL_LINE \
129 "Total: %d processes, %d lwps, load averages: %3.2f, %3.2f, %3.2f"

131 /* global variables */

133 static char      *t_ulon;          /* termcap: start underline */
134 static char      *t_uloff;        /* termcap: end underline */
135 static char      *t_up;           /* termcap: cursor 1 line up */
136 static char      *t_eol;         /* termcap: clear end of line */
137 static char      *t_smcup;       /* termcap: cursor mvcp on */
138 static char      *t_rmcup;       /* termcap: cursor mvcp off */
139 static char      *t_home;        /* termcap: move cursor home */
140 static char      *movecur = NULL; /* termcap: move up string */
141 static char      *empty_string = "\0"; /* termcap: empty string */
142 static uint_t    print_movecur = FALSE; /* print movecur or not */
143 static int       is_curses_on = FALSE; /* current curses state */

145 static table_t   pid_tbl = {0, 0, NULL}; /* selected processes */
146 static table_t   cpu_tbl = {0, 0, NULL}; /* selected processors */
147 static table_t   set_tbl = {0, 0, NULL}; /* selected processor sets */
148 static table_t   prj_tbl = {0, 0, NULL}; /* selected projects */
149 static table_t   tsk_tbl = {0, 0, NULL}; /* selected tasks */
150 static table_t   lgr_tbl = {0, 0, NULL}; /* selected lgroups */
151 static zonetbl_t zone_tbl = {0, 0, NULL}; /* selected zones */
152 static uidtbl_t  euiddtbl = {0, 0, NULL}; /* selected effective users */
153 static uidtbl_t  ruidtbl = {0, 0, NULL}; /* selected real users */

155 static uint_t    total_procs;      /* total number of procs */
156 static uint_t    total_lwps;      /* total number of lwps */
157 static float     total_cpu;       /* total cpu usage */
158 static float     total_mem;       /* total memory usage */

160 static list_t    lwps;            /* list of lwps/processes */
161 static list_t    users;          /* list of users */
162 static list_t    tasks;         /* list of tasks */
163 static list_t    projects;      /* list of projects */
164 static list_t    zones;         /* list of zones */
165 static list_t    lgroups;       /* list of lgroups */

167 static volatile uint_t sigwinch = 0;
168 static volatile uint_t sigtstp = 0;
169 static volatile uint_t sigterm = 0;

171 static long pagesize;

173 /* default settings */

175 static optdesc_t opts = {
176     5,          /* interval between updates, seconds */
177     15,        /* number of lines in top part */
178     5,         /* number of lines in bottom part */
179     -1,       /* number of iterations; infinitely */
180     OPT_PSINFO | OPT_FULLSCREEN | OPT_USEHOME | OPT_TERMCAP,
181     -1        /* sort in decreasing order */
182 };

```

unchanged portion omitted

```

350 /*
351  * A routine to display the contents of the list on the screen
352  */
353 static void
354 list_print(list_t *list)
355 {

```

```

356     lwp_info_t *lwp;
357     id_info_t *id;
358     char usr[4], sys[4], trp[4], tfl[4];
359     char dfl[4], lck[4], slp[4], lat[4];
360     char vcx[4], icx[4], scl[4], sig[4];
361     char psize[6], prssize[6], pmem[6], pcpu[6], ptime[12];
362     char pstate[7], pnice[4], ppri[4];
363     char pname[LOGNAME_MAX+1];
364     char projname[PROJNAME_MAX+1];
365     char zonename[ZONENAME_MAX+1];
366     float cpu, mem;
367     double loadavg[3] = {0, 0, 0};
368     int i, lwpid;

370     if (foreach_element(&set_tbl, &loadavg, psetloadavg) == 0) {
371         /*
372          * If processor sets aren't specified, we display system-wide
373          * load averages.
374          */
375         (void) getloadavg(loadavg, 3);
376     }

378     if (((opts.o_outpmode & OPT_UPDATE) || (opts.o_outpmode & OPT_DDATE)) &&
379         ((list->l_type == LT_LWPS) || !(opts.o_outpmode & OPT_SPLIT)))
380         print_timestamp();
381     if (opts.o_outpmode & OPT_TTY)
382         (void) putchar('\r');
383     (void) putp(t_ulon);

385     switch (list->l_type) {
386     case LT_PROJECTS:
387         if (opts.o_outpmode & OPT_LWPS)
388             (void) printf(PROJECT_HEADER_LWP);
389         else
390             (void) printf(PROJECT_HEADER_PROC);
391         break;
392     case LT_TASKS:
393         if (opts.o_outpmode & OPT_LWPS)
394             (void) printf(TASK_HEADER_LWP);
395         else
396             (void) printf(TASK_HEADER_PROC);
397         break;
398     case LT_ZONES:
399         if (opts.o_outpmode & OPT_LWPS)
400             (void) printf(ZONE_HEADER_LWP);
401         else
402             (void) printf(ZONE_HEADER_PROC);
403         break;
404     case LT_USERS:
405         if (opts.o_outpmode & OPT_LWPS)
406             (void) printf(USER_HEADER_LWP);
407         else
408             (void) printf(USER_HEADER_PROC);
409         break;
410     case LT_LWPS:
411         if (opts.o_outpmode & OPT_LWPS) {
412             if (opts.o_outpmode & OPT_PSINFO) {
413                 if (opts.o_outpmode & OPT_LGRP)
414                     (void) printf(PSINFO_HEADER_LWP_LGRP);
415                 else
416                     (void) printf(PSINFO_HEADER_LWP);
417             }
418             if (opts.o_outpmode & OPT_MSACCT)
419                 (void) printf(USAGE_HEADER_LWP);
420         } else {
421             if (opts.o_outpmode & OPT_PSINFO) {

```

```

422         if (opts.o_outpmode & OPT_LGRP)
423             (void) printf(PSINFO_HEADER_PROC_LGRP);
424         else
425             (void) printf(PSINFO_HEADER_PROC);
426     }
427     if (opts.o_outpmode & OPT_MSACCT)
428         (void) printf(USAGE_HEADER_PROC);
429     }
430     break;
431 }

433 (void) putp(t_uloff);
434 (void) putp(t_eol);
435 (void) putchar('\n');

437 for (i = 0; i < list->l_used; i++) {
438     switch (list->l_type) {
439     case LT_PROJECTS:
440     case LT_TASKS:
441     case LT_USERS:
442     case LT_ZONES:
443         id = list->l_ptrs[i];
444         /*
445          * CPU usage and memory usage normalization
446          */
447         if (total_cpu >= 100)
448             cpu = (100 * id->id_pctcpu) / total_cpu;
449         else
450             cpu = id->id_pctcpu;
451         if (id->id_sizematch == B_FALSE && total_mem >= 100)
452             mem = (100 * id->id_pctmem) / total_mem;
453         else
454             mem = id->id_pctmem;
455         if (list->l_type == LT_USERS) {
456             pwd_getname(id->id_uid, pname, sizeof (pname),
457                 opts.o_outpmode & OPT_NORESOLVE,
458                 opts.o_outpmode & (OPT_TERMCAP|OPT_TRUNC),
459                 LOGIN_WIDTH);
460         } else if (list->l_type == LT_ZONES) {
461             if (list->l_type == LT_USERS)
462                 pwd_getname(id->id_uid, pname, LOGNAME_MAX + 1,
463                     opts.o_outpmode & OPT_NORESOLVE);
464             else if (list->l_type == LT_ZONES)
465                 getzonename(id->id_zoneid, zonename,
466                     sizeof (zonename),
467                     opts.o_outpmode & (OPT_TERMCAP|OPT_TRUNC),
468                     ZONE_WIDTH);
469         } else {
470             ZONENAME_MAX;
471         } else
472             else
473                 getprojname(id->id_projid, projname,
474                     sizeof (projname),
475                     opts.o_outpmode & OPT_NORESOLVE,
476                     opts.o_outpmode & (OPT_TERMCAP|OPT_TRUNC),
477                     PROJECT_WIDTH);
478         }
479         PROJNAME_MAX,
480         opts.o_outpmode & OPT_NORESOLVE);
481     Format_size(psize, id->id_size, 6);
482     Format_size(prssize, id->id_rssize, 6);
483     Format_pct(pmem, mem, 4);
484     Format_pct(pcpu, cpu, 4);
485     Format_time(pptime, id->id_time, 10);
486     if (opts.o_outpmode & OPT_TTY)
487         (void) putchar('\r');
488     if (list->l_type == LT_PROJECTS)

```

```

480         (void) printf(PROJECT_LINE, (int)id->id_projid,
481             id->id_nproc, psize, prssize, pmem, ptime,
482             pcpu, projname);
483     else if (list->l_type == LT_TASKS)
484         (void) printf(TASK_LINE, (int)id->id_taskid,
485             id->id_nproc, psize, prssize, pmem, ptime,
486             pcpu, projname);
487     else if (list->l_type == LT_ZONES)
488         (void) printf(ZONE_LINE, (int)id->id_zoneid,
489             id->id_nproc, psize, prssize, pmem, ptime,
490             pcpu, zonename);
491     else
492         (void) printf(USER_LINE, id->id_nproc, pname,
493             psize, prssize, pmem, ptime, pcpu);
494     (void) putp(t_eol);
495     (void) putchar('\n');
496     break;
497 case LT_LWPS:
498     lwp = list->l_ptrs[i];
499     if (opts.o_outpmode & OPT_LWPS)
500         lwpid = lwp->li_info.pr_lwp.pr_lwpid;
501     else
502         lwpid = lwp->li_info.pr_nlwp +
503             lwp->li_info.pr_nzomb;
504     pwd_getname(lwp->li_info.pr_uid, pname, sizeof (pname),
505         opts.o_outpmode & OPT_NORESOLVE,
506         opts.o_outpmode & (OPT_TERMCAP|OPT_TRUNC),
507         LOGIN_WIDTH);
508     pwd_getname(lwp->li_info.pr_uid, pname, LOGNAME_MAX + 1,
509         opts.o_outpmode & OPT_NORESOLVE);
510     if (opts.o_outpmode & OPT_PSINFO) {
511         Format_size(psize, lwp->li_info.pr_size, 6);
512         Format_size(prssize, lwp->li_info.pr_rssize, 6);
513         Format_state(pstate,
514             lwp->li_info.pr_lwp.pr_sname,
515             lwp->li_info.pr_lwp.pr_onpro, 7);
516         if (strcmp(lwp->li_info.pr_lwp.pr_clname,
517             "RT") == 0 ||
518             strcmp(lwp->li_info.pr_lwp.pr_clname,
519             "SYS") == 0 ||
520             lwp->li_info.pr_lwp.pr_sname == 'Z')
521             (void) strcpy(pnice, " -");
522     } else
523         Format_num(pnice,
524             lwp->li_info.pr_lwp.pr_nice - NZERO,
525             4);
526     Format_num(ppri, lwp->li_info.pr_lwp.pr_pri, 4);
527     Format_pct(pcpu,
528         FRC2PCT(lwp->li_info.pr_lwp.pr_pctcpu), 4);
529     if (opts.o_outpmode & OPT_LWPS)
530         Format_time(pptime,
531             lwp->li_info.pr_lwp.pr_time.tv_sec,
532             10);
533     else
534         Format_time(pptime,
535             lwp->li_info.pr_time.tv_sec, 10);
536     if (opts.o_outpmode & OPT_TTY)
537         (void) putchar('\r');
538     stripfname(lwp->li_info.pr_fname);
539     if (opts.o_outpmode & OPT_LGRP) {
540         (void) printf(PSINFO_LINE_LGRP,
541             (int)lwp->li_info.pr_pid, pname,
542             psize, prssize, pstate,
543             ppri, pnice, ptime, pcpu,
544             psize, prssize, pstate, ppri, pnice,
545             ptime, pcpu,

```

```

542         (int)lwp->li_info.pr_lwp.pr_lgrp,
543         lwp->li_info.pr_fname, lwpid);
544     } else {
545         (void) printf(PSINFO_LINE,
546         (int)lwp->li_info.pr_pid, pname,
547         psize, prssize,
548         pstate, ppri, pnice,
549         psize, prssize, pstate, ppri, pnice,
550         ptime, pcpu,
551         lwp->li_info.pr_fname, lwpid);
552     }
553     (void) putp(t_eol);
554     (void) putchar('\n');
555 }
556 if (opts.o_outpmode & OPT_MSACCT) {
557     Format_pct(usr, lwp->li_usr, 4);
558     Format_pct(sys, lwp->li_sys, 4);
559     Format_pct(slp, lwp->li_slp, 4);
560     Format_num(vcx, lwp->li_vcx, 4);
561     Format_num(icx, lwp->li_icx, 4);
562     Format_num(scl, lwp->li_scl, 4);
563     Format_num(sig, lwp->li_sig, 4);
564     Format_pct(trp, lwp->li_trp, 4);
565     Format_pct(tfl, lwp->li_tfl, 4);
566     Format_pct(dfl, lwp->li_dfl, 4);
567     Format_pct(lck, lwp->li_lck, 4);
568     Format_pct(lat, lwp->li_lat, 4);
569     if (opts.o_outpmode & OPT_TTY)
570         (void) putchar('\r');
571     stripfname(lwp->li_info.pr_fname);
572     (void) printf(USAGE_LINE,
573     (int)lwp->li_info.pr_pid, pname,
574     usr, sys, trp, tfl, dfl, lck,
575     slp, lat, vcx, icx, scl, sig,
576     lwp->li_info.pr_fname, lwpid);
577     (void) putp(t_eol);
578     (void) putchar('\n');
579 }
580 break;
581 }
582 }
583 if (opts.o_outpmode & OPT_TTY)
584     (void) putchar('\r');
585 if (opts.o_outpmode & OPT_TERMCAP) {
586     switch (list->l_type) {
587     case LT_PROJECTS:
588     case LT_USERS:
589     case LT_TASKS:
590     case LT_ZONES:
591         while (i++ < opts.o_nbottom) {
592             (void) putp(t_eol);
593             (void) putchar('\n');
594         }
595         break;
596     case LT_LWPS:
597         while (i++ < opts.o_ntop) {
598             (void) putp(t_eol);
599             (void) putchar('\n');
600         }
601     }
602 }
603 }
604 if (opts.o_outpmode & OPT_TTY)
605     (void) putchar('\r');

```

```

607     if ((opts.o_outpmode & OPT_SPLIT) && list->l_type == LT_LWPS)
608         return;
609 }
610 (void) printf(TOTAL_LINE, total_procs, total_lwps,
611 loadavg[LOADAVG_1MIN], loadavg[LOADAVG_5MIN],
612 loadavg[LOADAVG_15MIN]);
613 (void) putp(t_eol);
614 (void) putchar('\n');
615 if (opts.o_outpmode & OPT_TTY)
616     (void) putchar('\r');
617 (void) putp(t_eol);
618 (void) fflush(stdout);
619 }

```

unchanged\_portion\_omitted

```

1363 int
1364 main(int argc, char **argv)
1365 {
1366     DIR *procdir;
1367     char *p;
1368     char *sortk = "cpu"; /* default sort key */
1369     int opt;
1370     int timeout;
1371     struct pollfd pollset;
1372     char key;
1373 }
1374 (void) setlocale(LC_ALL, "");
1375 (void) textdomain(TEXT_DOMAIN);
1376 Progname(argv[0]);
1377 lwpid_init();
1378 fd_init(Setrlimit());
1379 }
1380 pagesize = sysconf(_SC_PAGESIZE);
1381 }
1382 while ((opt = getopt(argc, argv,
1383 "vcd:HmarRLtu:U:n:p:C:P:h:s:S:j:k:TJwz:Z")) != (int)EOF) {
1384     switch (opt) {
1385     case 'r':
1386         opts.o_outpmode |= OPT_NORESOLVE;
1387         break;
1388     case 'R':
1389         opts.o_outpmode |= OPT_REALTIME;
1390         break;
1391     case 'c':
1392         opts.o_outpmode &= ~OPT_TERMCAP;
1393         opts.o_outpmode &= ~OPT_FULLSCREEN;
1394         break;
1395     case 'd':
1396         if (optarg) {
1397             if (*optarg == 'u')
1398                 opts.o_outpmode |= OPT_UPDATE;
1399             else if (*optarg == 'd')
1400                 opts.o_outpmode |= OPT_DDATE;
1401             else
1402                 Usage();
1403         } else {
1404             Usage();
1405         }
1406     }
1407     break;
1408 case 'h':
1409     fill_table(&lgr_tbl, optarg, 'h');
1410     break;
1411 case 'H':
1412     opts.o_outpmode |= OPT_LGRP;

```

```

1412         break;
1413     case 'm':
1414     case 'v':
1415         opts.o_outpmode &= ~OPT_PSINFO;
1416         opts.o_outpmode |= OPT_MSACCT;
1417         break;
1418     case 't':
1419         opts.o_outpmode &= ~OPT_PSINFO;
1420         opts.o_outpmode |= OPT_USERS;
1421         break;
1422     case 'a':
1423         opts.o_outpmode |= OPT_SPLIT | OPT_USERS;
1424         break;
1425     case 'T':
1426         opts.o_outpmode |= OPT_SPLIT | OPT_TASKS;
1427         break;
1428     case 'J':
1429         opts.o_outpmode |= OPT_SPLIT | OPT_PROJECTS;
1430         break;
1431     case 'n':
1432         if ((p = strtok(optarg, ",") == NULL)
1433             Die(gettext("invalid argument for -n\n"));
1434         opts.o_ntop = Atoi(p);
1435         if (p = strtok(NULL, ",")
1436             opts.o_nbottom = Atoi(p);
1437         opts.o_outpmode &= ~OPT_FULLSCREEN;
1438         break;
1439     case 's':
1440         opts.o_sortorder = -1;
1441         sortk = optarg;
1442         break;
1443     case 'S':
1444         opts.o_sortorder = 1;
1445         sortk = optarg;
1446         break;
1447     case 'u':
1448         if ((p = strtok(optarg, ",") == NULL)
1449             Die(gettext("invalid argument for -u\n"));
1450         add_uid(&euid_tbl, p);
1451         while (p = strtok(NULL, ",")
1452             add_uid(&euid_tbl, p);
1453         break;
1454     case 'U':
1455         if ((p = strtok(optarg, ",") == NULL)
1456             Die(gettext("invalid argument for -U\n"));
1457         add_uid(&ruid_tbl, p);
1458         while (p = strtok(NULL, ",")
1459             add_uid(&ruid_tbl, p);
1460         break;
1461     case 'p':
1462         fill_table(&pid_tbl, optarg, 'p');
1463         break;
1464     case 'C':
1465         fill_set_table(optarg);
1466         opts.o_outpmode |= OPT_PSETS;
1467         break;
1468     case 'P':
1469         fill_table(&cpu_tbl, optarg, 'P');
1470         break;
1471     case 'k':
1472         fill_table(&tsk_tbl, optarg, 'k');
1473         break;
1474     case 'j':
1475         fill_prj_table(optarg);
1476         break;
1477     case 'L':

```

```

1478         opts.o_outpmode |= OPT_LWPS;
1479         break;
1480     case 'W':
1481         opts.o_outpmode |= OPT_TRUNC;
1482         break;
1483     case 'z':
1484         if ((p = strtok(optarg, ",") == NULL)
1485             Die(gettext("invalid argument for -z\n"));
1486         add_zone(&zone_tbl, p);
1487         while (p = strtok(NULL, ",")
1488             add_zone(&zone_tbl, p);
1489         break;
1490     case 'Z':
1491         opts.o_outpmode |= OPT_SPLIT | OPT_ZONES;
1492         break;
1493     default:
1494         Usage();
1495     }
1496 }
1497
1498 (void) atexit(Exit);
1499 if ((opts.o_outpmode & OPT_USERS) &&
1500     !(opts.o_outpmode & OPT_SPLIT))
1501     opts.o_nbottom = opts.o_ntop;
1502 if (opts.o_ntop == 0 || opts.o_nbottom == 0)
1503     Die(gettext("invalid argument for -n\n"));
1504 if (!(opts.o_outpmode & OPT_SPLIT) && (opts.o_outpmode & OPT_USERS) &&
1505     ((opts.o_outpmode & (OPT_PSINFO | OPT_MSACCT)))
1506     Die(gettext("-t option cannot be used with -v or -m\n"));
1507
1508 if ((opts.o_outpmode & OPT_SPLIT) && (opts.o_outpmode & OPT_USERS) &&
1509     !(opts.o_outpmode & (OPT_PSINFO | OPT_MSACCT)))
1510     Die(gettext("-t option cannot be used with "
1511         "-a, -J, -T or -Z\n"));
1512
1513 if ((opts.o_outpmode & OPT_USERS) &&
1514     (opts.o_outpmode & (OPT_TASKS | OPT_PROJECTS | OPT_ZONES)))
1515     Die(gettext("-a option cannot be used with "
1516         "-t, -J, -T or -Z\n"));
1517
1518 if (((opts.o_outpmode & OPT_TASKS) &&
1519     (opts.o_outpmode & (OPT_PROJECTS|OPT_ZONES))) ||
1520     ((opts.o_outpmode & OPT_PROJECTS) &&
1521     (opts.o_outpmode & (OPT_TASKS|OPT_ZONES)))) {
1522     Die(gettext(
1523         "-J, -T and -Z options are mutually exclusive\n"));
1524 }
1525
1526 /*
1527 * There is not enough space to combine microstate information and
1528 * lgroup information and still fit in 80-column output.
1529 */
1530 if ((opts.o_outpmode & OPT_LGRP) && (opts.o_outpmode & OPT_MSACCT)) {
1531     Die(gettext("-H and -m options are mutually exclusive\n"));
1532 }
1533
1534 if (argc > optind)
1535     opts.o_interval = Atoi(argv[optind++]);
1536 if (argc > optind)
1537     opts.o_count = Atoi(argv[optind++]);
1538 if (opts.o_count == 0)
1539     Die(gettext("invalid counter value\n"));
1540 if (argc > optind)
1541     Usage();
1542 if (opts.o_outpmode & OPT_REALTIME)
1543     Prioctl("RT");

```

```

1544     if (isatty(STDOUT_FILENO) == 1 && isatty(STDIN_FILENO))
1545         opts.o_outpmode |= OPT_TTY;      /* interactive */
1546     if (!(opts.o_outpmode & OPT_TTY)) {
1547         opts.o_outpmode &= ~OPT_TERMCAP; /* no termcap for pipes */
1548         opts.o_outpmode &= ~OPT_FULLSCREEN;
1549     }
1550     if (opts.o_outpmode & OPT_TERMCAP)
1551         ldtermcap(); /* can turn OPT_TERMCAP off */
1552     if (opts.o_outpmode & OPT_TERMCAP)
1553         (void) setsize();
1554     list_alloc(&lwps, opts.o_ntop);
1555     list_alloc(&users, opts.o_nbottom);
1556     list_alloc(&tasks, opts.o_nbottom);
1557     list_alloc(&projects, opts.o_nbottom);
1558     list_alloc(&zones, opts.o_nbottom);
1559     list_alloc(&lggroups, opts.o_nbottom);
1560     list_setkeyfunc(sortk, &opts, &lwps, LT_LWPS);
1561     list_setkeyfunc(NULL, &opts, &users, LT_USERS);
1562     list_setkeyfunc(NULL, &opts, &tasks, LT_TASKS);
1563     list_setkeyfunc(NULL, &opts, &projects, LT_PROJECTS);
1564     list_setkeyfunc(NULL, &opts, &zones, LT_ZONES);
1565     list_setkeyfunc(NULL, &opts, &lggroups, LT_LGRPS);
1566     if (opts.o_outpmode & OPT_TERMCAP)
1567         curses_on();
1568     if ((procdir = opendir("/proc")) == NULL)
1569         Die(gettext("cannot open /proc directory\n"));
1570     if (opts.o_outpmode & OPT_TTY) {
1571         (void) printf(gettext("Please wait...\r"));
1572         if (!(opts.o_outpmode & OPT_TERMCAP))
1573             (void) putchar('\n');
1574         (void) fflush(stdout);
1575     }
1576     set_signals();
1577     pollset.fd = STDIN_FILENO;
1578     pollset.events = POLLIN;
1579     timeout = opts.o_interval * MILLISEC;

1581     /*
1582     * main program loop
1583     */
1584     do {
1585         if (sigterm == 1)
1586             break;
1587         if (sigtstp == 1) {
1588             curses_off();
1589             (void) signal(SIGTSTP, SIG_DFL);
1590             (void) kill(0, SIGTSTP);
1591             /*
1592             * prstat stops here until it receives SIGCONT signal.
1593             */
1594             sigtstp = 0;
1595             (void) signal(SIGTSTP, sig_handler);
1596             curses_on();
1597             print_movecur = FALSE;
1598             if (opts.o_outpmode & OPT_FULLSCREEN)
1599                 sigwinch = 1;
1600         }
1601         if (sigwinch == 1) {
1602             if (setsize() == 1) {
1603                 list_free(&lwps);
1604                 list_free(&users);
1605                 list_free(&tasks);
1606                 list_free(&projects);
1607                 list_free(&zones);
1608                 list_alloc(&lwps, opts.o_ntop);
1609                 list_alloc(&users, opts.o_nbottom);

```

```

1610         list_alloc(&tasks, opts.o_nbottom);
1611         list_alloc(&projects, opts.o_nbottom);
1612         list_alloc(&zones, opts.o_nbottom);
1613     }
1614     sigwinch = 0;
1615     (void) signal(SIGWINCH, sig_handler);
1616 }
1617 prstat_scandir(procdir);
1618 list_refresh(&lwps);
1619 if (print_movecur)
1620     (void) putp(movecur);
1621 print_movecur = TRUE;
1622 if ((opts.o_outpmode & OPT_PSINFO) ||
1623     (opts.o_outpmode & OPT_MSACCT)) {
1624     list_sort(&lwps);
1625     list_print(&lwps);
1626 }
1627 if (opts.o_outpmode & OPT_USERS) {
1628     list_getsize(&users);
1629     list_sort(&users);
1630     list_print(&users);
1631     list_clear(&users);
1632 }
1633 if (opts.o_outpmode & OPT_TASKS) {
1634     list_getsize(&tasks);
1635     list_sort(&tasks);
1636     list_print(&tasks);
1637     list_clear(&tasks);
1638 }
1639 if (opts.o_outpmode & OPT_PROJECTS) {
1640     list_getsize(&projects);
1641     list_sort(&projects);
1642     list_print(&projects);
1643     list_clear(&projects);
1644 }
1645 if (opts.o_outpmode & OPT_ZONES) {
1646     list_getsize(&zones);
1647     list_sort(&zones);
1648     list_print(&zones);
1649     list_clear(&zones);
1650 }
1651 if (opts.o_count == 1)
1652     break;
1653 /*
1654 * If poll() returns -1 and sets errno to EINTR here because
1655 * the process received a signal, it is Ok to abort this
1656 * timeout and loop around because we check the signals at the
1657 * top of the loop.
1658 */
1659 if (opts.o_outpmode & OPT_TTY) {
1660     if (poll(&pollset, (nfds_t)1, timeout) > 0) {
1661         if (read(STDIN_FILENO, &key, 1) == 1) {
1662             if (tolower(key) == 'q')
1663                 break;
1664         }
1665     }
1666     } else {
1667         (void) sleep(opts.o_interval);
1668     }
1669 } while (opts.o_count == (-1) || --opts.o_count);

1671     if (opts.o_outpmode & OPT_TTY)
1672         (void) putchar('\r');
1673     return (0);
1674 }

```

unchanged portion omitted



new/usr/src/cmd/prstat/prstat.h

1

```
*****
5877 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/prstat/prstat.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
25 * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
26 * Use is subject to license terms.
27 *
28 * Portions Copyright 2009 Chad Mynhier
29 */
31 #ifndef _PRSTAT_H
32 #define _PRSTAT_H
34 #include <sys/sysmacros.h>
35 #include <sys/time.h>
36 #include <sys/types.h>
37 #include <procfs.h>
39 #ifdef __cplusplus
40 extern "C" {
41 #endif
43 /*
44  * FRC2PCT macro is used to convert 16-bit binary fractions in the range
45  * 0.0 to 1.0 with binary point to the right of the high order bit
46  * (i.e. 1.0 == 0x8000) to percentage value.
47  */
49 #define FRC2PCT(pp)      (((float)(pp))/0x8000*100)
51 #define TIME2NSEC(__t)\
52 (hrtime_t)(((hrtime_t)__t.tv_sec * (hrtime_t)NANOSEC) + (hrtime_t)__t.tv_nsec)
53 #define TIME2SEC(__t)\
54 (hrtime_t)(__t.tv_sec)
56 /*
57  * List of available output modes
58  */
59 #define OPT_PSINFO      0x0001      /* read process's data from "psinfo" */
60 #define OPT_LWPS       0x0002      /* report about all lwps */
```

new/usr/src/cmd/prstat/prstat.h

2

```
61 #define OPT_USERS      0x0004      /* report about most active users */
62 #define OPT_UNUSED    0x0008      /* reserved for future use */
63 #define OPT_REALTIME  0x0010      /* real-time scheduling class flag */
64 #define OPT_MSACCT    0x0020      /* microstate accounting flag */
65 #define OPT_TERMCAP   0x0040      /* use termcap data to move cursor */
66 #define OPT_SPLIT    0x0080      /* split-screen mode flag */
67 #define OPT_TTY      0x0100      /* report results to tty or file */
68 #define OPT_FULLSCREEN 0x0200      /* full-screen mode flag */
69 #define OPT_USEHOME   0x0400      /* use 'home' to move cursor up */
70 #define OPT_TASKS     0x0800      /* report about system tasks */
71 #define OPT_PROJECTS  0x1000      /* report about system projects */
72 #define OPT_ZONES     0x2000      /* report about zones */
73 #define OPT_PSETS     0x4000      /* report for specified psets */
74 #define OPT_LGRP      0x8000      /* report home lgroups */
75 #define OPT_UPDATE    0x20000     /* print unix timestamp */
76 #define OPT_DDATE     0x40000     /* print timestamp in date(1) format */
77 #define OPT_NORESOLVE 0x80000     /* no nsswitch lookups */
78 #define OPT_TRUNC     0x100000    /* truncate long names */
80 /*
81  * Flags to keep track of process or lwp status
82  */
83 #define LWP_ALIVE      0x0008      /* this pid/lwp still exists */
84 #define LWP_REPRESENT 0x0010      /* this LWP represents the process */
86 /*
87  * Possible list types
88  */
89 #define LT_LWPS        0x0001
90 #define LT_USERS       0x0002
91 #define LT_TASKS       0x0004
92 #define LT_PROJECTS    0x0008
93 #define LT_ZONES       0x0010
94 #define LT_LGRPS       0x0020
96 /*
97  * Linked list of per-process or per-lwp statistics
98  */
99 typedef struct lwp_info {
100     psinfo_t      li_info;      /* data read from psinfo file */
101     prusage_t     li_usage;     /* data read from usage file */
102     ulong_t       li_key;       /* value of the key for this lwp */
103     int           li_flags;     /* process/lwp flags */
104     float         li_usr;       /* user level CPU time */
105     float         li_sys;       /* system call CPU time */
106     float         li_trp;       /* other system trap CPU time */
107     float         li_tfl;       /* text page fault sleep time */
108     float         li_dfl;       /* data page fault sleep time */
109     float         li_lck;       /* user lock wait sleep time */
110     float         li_slp;       /* all other sleep time */
111     float         li_lat;       /* wait-cpu (latency) time */
112     ulong_t       li_vcx;       /* voluntary context switches */
113     ulong_t       li_icx;       /* involuntary context switches */
114     ulong_t       li_scl;       /* system calls */
115     ulong_t       li_sig;       /* received signals */
116     struct lwp_info *li_next;   /* pointer to next lwp */
117     struct lwp_info *li_prev;   /* pointer to previous lwp */
118 } lwp_info_t;
_____
unchanged portion omitted
```

```

*****
6847 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/prstat/prtable.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  *
27  * Portions Copyright 2009 Chad Mynhier
28  */

30 #include <proofs.h>
31 #include <unistd.h>
32 #include <stdlib.h>
33 #include <pwd.h>
34 #include <ctype.h>
35 #include <string.h>
36 #include <libintl.h>
37 #include <errno.h>
38 #include <zone.h>
39 #include <libzonecfg.h>

41 #include "prstat.h"
42 #include "prutil.h"
43 #include "prtable.h"

45 static plwp_t *plwp_tbl[PLWP_TBL_SZ];

47 void
48 lwpid_init()
49 {
50     (void) memset(&plwp_tbl, 0, sizeof (plwp_t *) * PLWP_TBL_SZ);
51 }
    unchanged_portion_omitted_

63 void
64 pwd_getname(uid_t uid, char *name, size_t length, int noresolve,
65             int trunc, size_t width)
66 {
67     struct passwd *pwd;
68     size_t n;

```

```

70     if (noresolve || (pwd = getpwuid(uid)) == NULL) {
71         n = snprintf(NULL, 0, "%u", uid);
72         if (trunc && n > width)
73             (void) snprintf(name, length, "%.u%c",
74                             width - 1, uid, '*');
75     } else
76         (void) snprintf(name, length, "%u", uid);
77     } else {
78         n = strlen(pwd->pw_name);
79         if (trunc && n > width)
80             (void) snprintf(name, length, "%.s%c",
81                             width - 1, pwd->pw_name, '*');
82     } else
83         (void) snprintf(name, length, "%s", pwd->pw_name);
84     }
85 }
    unchanged_portion_omitted_

```

new/usr/src/cmd/prstat/prtable.h

1

```
*****
2466 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/prstat/prtable.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  *
27  * Portions Copyright 2009 Chad Mynhier
28  */

30 #ifndef _PRTABLE_H
31 #define _PRTABLE_H

33 #ifdef __cplusplus
34 extern "C" {
35 #endif

37 #include <limits.h>
38 #include <zone.h>
39 #include "prstat.h"

41 #define PLWP_TBL_SZ 4096 /* hash table of plwp_t structures */
42 #define LWP_ACTIVE 1

44 typedef struct {
45     size_t t_size;
46     size_t t_nent;
47     long *t_list;
48 } table_t;
    unchanged_portion_omitted

75 extern void pwd_getname(uid_t, char *, size_t, int, int, size_t);
73 extern void pwd_getname(uid_t, char *, int, int);
76 extern void add_uid(uidtbl_t *, char *);
77 extern int has_uid(uidtbl_t *, uid_t);
78 extern void add_element(table_t *, long);
79 extern int has_element(table_t *, long);
80 extern void add_zone(zonetbl_t *, char *);
81 extern int has_zone(zonetbl_t *, zoneid_t);
82 extern void convert_zone(zonetbl_t *);
83 extern int foreach_element(table_t *, void *, void (*)(long, void *));
```

new/usr/src/cmd/prstat/prtable.h

2

```
84 extern void lwpid_init();
85 extern void lwpid_add(lwp_info_t *, pid_t, id_t);
86 extern lwp_info_t *lwpid_get(pid_t, id_t);
87 extern int lwpid_pidcheck(pid_t);
88 extern void lwpid_del(pid_t, id_t);
89 extern void lwpid_set_active(pid_t, id_t);
90 extern int lwpid_is_active(pid_t, id_t);

92 #ifdef __cplusplus
93 }
    unchanged_portion_omitted
```

new/usr/src/cmd/prstat/prutil.c

1

```
*****
7544 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/prstat/prutil.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
24 * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25 * Use is subject to license terms.
26 *
27 * Portions Copyright 2009 Chad Mynhier
28 */

30 #include <sys/types.h>
31 #include <sys/param.h>
32 #include <sys/resource.h>
33 #include <sys/priocntl.h>
34 #include <sys/rtpriocntl.h>
35 #include <sys/tspriocntl.h>
36 #include <zone.h>

38 #include <libintl.h>
39 #include <limits.h>
40 #include <wchar.h>
41 #include <unistd.h>
42 #include <string.h>
43 #include <stdlib.h>
44 #include <stdarg.h>
45 #include <stdio.h>
46 #include <stdio_ext.h>
47 #include <errno.h>
48 #include <ctype.h>
49 #include <poll.h>
50 #include <project.h>

52 #include "prfile.h"
53 #include "prstat.h"
54 #include "prutil.h"

56 static char PRG_FMT[] = "%s: ";
57 static char ERR_FMT[] = ": %s\n";
58 static char *progname;
59 static char projbuf[PROJECT_BUFSZ];
```

new/usr/src/cmd/prstat/prutil.c

2

```
61 #define RLIMIT_NOFILE_MAX 32767

63 /*PRINTFLIKE1*/
64 void
65 Warn(char *format, ...)
66 {
67     int err = errno;
68     va_list alist;

70     if (progname != NULL)
71         (void) fprintf(stderr, PRG_FMT, progname);
72     va_start(alist, format);
73     (void) vfprintf(stderr, format, alist);
74     va_end(alist);
75     if (strchr(format, '\n') == NULL)
76         (void) fprintf(stderr, gettext(ERR_FMT), strerror(err));
77 }
    unchanged_portion_omitted

107 void
108 Usage()
109 {
110     (void) fprintf(stderr, gettext(
111         "Usage:\tprstat [-achJLmrRtTvWZ] [-u eidlist] [-U uidlist]\n"
112         "Usage:\tprstat [-achJLmrRtTvZ] [-u eidlist] [-U uidlist]\n"
113         "\t[-p pidlist] [-P cpulist] [-C prsetlist] [-h lgroupplist]\n"
114         "\t[-j projidlist] [-k taskidlist] [-z zoneidlist]\n"
115         "\t[-s key | -S key] [-n nprocs[,nusers]] [-d d|u]\n"
116         "\t[interval [counter]]\n"));
117 }
    unchanged_portion_omitted

282 void
283 getprojname(projid_t projid, char *str, size_t len, int noresolve,
284             int trunc, size_t width)
285 getprojname(projid_t projid, char *str, int len, int noresolve)
286 {
287     struct project proj;
288     size_t n;

289     if (noresolve || getprojbyid(projid, &proj, projbuf, PROJECT_BUFSZ) ==
290         NULL) {
291         (void) snprintf(str, len, "%-6d", (int)projid);
292     } else {
293         n = strlen(proj.pj_name);
294         if (trunc && n > width)
295             (void) snprintf(str, len, "%.*s%c", width - 1,
296                             proj.pj_name, '*');
297         else
298             (void) snprintf(str, len, "%-28s", proj.pj_name);
299     }
300 }

302 void
303 getzonename(zoneid_t zoneid, char *str, size_t len, int trunc, size_t width)
304 getzonename(zoneid_t zoneid, char *str, int len)
305 {
306     char zone_name[ZONENAME_MAX];
307     size_t n;

308     if (getzonenamebyid(zoneid, zone_name, sizeof (zone_name)) < 0) {
309         if (getzonenamebyid(zoneid, zone_name, sizeof (zone_name)) < 0)
310             (void) snprintf(str, len, "%-6d", (int)zoneid);
311     } else {
```

```
311         n = strlen(zone_name);
312         if (trunc && n > width)
313             (void) snprintf(str, len, "%.*s%c", width - 1,
314                 zone_name, '*');
315         else
316             (void) snprintf(str, len, "%-28s", zone_name);
317     }
318 }
_____unchanged_portion_omitted_____
```

new/usr/src/cmd/prstat/prutil.h

1

\*\*\*\*\*

1878 Sat Jun 1 10:02:49 2013

new/usr/src/cmd/prstat/prutil.h

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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 *
24 * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25 * Use is subject to license terms.
26 *
27 * Portions Copyright 2009 Chad Mynhier
28 */

30 #ifndef _PRUTIL_H
31 #define _PRUTIL_H

33 #include <sys/processor.h>
34 #include <sys/types.h>

36 #ifdef __cplusplus
37 extern "C" {
38 #endif

40 extern void Die(char *, ...);
41 extern void Warn(char *, ...);
42 extern void Progname(char *);
43 extern void Usage();
44 extern int Atoi(char *);
45 extern void Format_size(char *, size_t, int);
46 extern void Format_pct(char *, float, int);
47 extern void Format_num(char *, int, int);
48 extern void Format_time(char *, ulong_t, int);
49 extern void Format_state(char *, char, processorid_t, int);
50 extern void *Realloc(void *, size_t);
51 extern void *Malloc(size_t);
52 extern void *Zalloc(size_t);
53 extern int Setrlimit();
54 extern void Prioctl(char *);
55 extern void getprojname(projid_t, char *, size_t, int, int, size_t);
56 extern void getzonename(projid_t, char *, size_t, int, size_t);
53 extern void getprojname(projid_t, char *, int, int);
54 extern void getzonename(projid_t, char *, int);
57 extern void stripfname(char *);
```

new/usr/src/cmd/prstat/prutil.h

2

```
59 #ifdef __cplusplus
60 }
_____unchanged_portion_omitted_
```

```

*****
60449 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/ps/ps.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
25 * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
26 * Use is subject to license terms.
27 */

29 /*
30 * Copyright (c) 2012, Joyent, Inc. All rights reserved.
31 */

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

36 /*
37  * ps -- print things about processes.
38  */
39 #include <stdio.h>
40 #include <ctype.h>
41 #include <string.h>
42 #include <errno.h>
43 #include <fcntl.h>
44 #include <pwd.h>
45 #include <grp.h>
46 #include <sys/types.h>
47 #include <sys/stat.h>
48 #include <sys/mkdev.h>
49 #include <unistd.h>
50 #include <stdlib.h>
51 #include <limits.h>
52 #include <dirent.h>
53 #include <sys/signal.h>
54 #include <sys/fault.h>
55 #include <sys/syscall.h>
56 #include <sys/time.h>
57 #include <procfs.h>
58 #include <locale.h>
59 #include <wctype.h>
60 #include <wchar.h>

```

```

61 #include <libw.h>
62 #include <stdarg.h>
63 #include <sys/proc.h>
64 #include <sys/pset.h>
65 #include <project.h>
66 #include <zone.h>

68 #define min(a, b)      ((a) > (b) ? (b) : (a))
69 #define max(a, b)      ((a) < (b) ? (b) : (a))

71 #define NTTY5         20      /* initial size of table for -t option */
72 #define SIZ           30      /* initial size of tables for -p, -s, -g, -h and -z */

74 /*
75  * Size of buffer holding args for t, p, s, g, u, U, G, z options.
76  * Set to ZONENAME_MAX, the minimum value needed to allow any
77  * zone to be specified.
78  */
79 #define ARGSIZ ZONENAME_MAX

81 #define MAXUGNAME (LOGNAME_MAX+2) /* max chars in a user/group */
82                                 /* name or printed u/g id */
83 #define MAXUGNAME 10 /* max chars in a user/group name or printed u/g id */

84 /* Structure for storing user or group info */
85 struct ugdata {
86     id_t      id; /* numeric user-id or group-id */
87     char      name[MAXUGNAME+1]; /* user/group name, null terminated */
88 };
89 #define unchanged_portion_omitted

218 #define NFIELDS (sizeof (fname) / sizeof (fname[0]))

220 static int      retcode = 1;
221 static int      lflg;
222 static int      aflg;
223 static int      uflg;
224 static int      Uflg;
225 static int      Gflg;
226 static int      aflg;
227 static int      dflg;
228 static int      Lflg;
229 static int      Pflg;
230 static int      Wflg;
231 static int      yflg;
232 static int      pflg;
233 static int      fflg;
234 static int      cflg;
235 static int      jflg;
236 static int      gflg;
237 static int      sflg;
238 static int      tflg;
239 static int      zflg;
240 static int      Zflg;
241 static int      hflg;
242 static int      Hflg;
243 static uid_t    tuid = (uid_t)-1;
244 static int      errflg;

246 static int      ndev; /* number of devices */
247 static int      maxdev; /* number of devl structures allocated */

249 #define DNINCR 100
250 #define DNSIZE 14
251 static struct devl { /* device list */
252     char      dname[DNSIZE]; /* device name */

```

```

253     dev_t   ddev;           /* device number */
254 } *devl;
    unchanged_portion_omitted

374 static int
375 stdmain(int argc, char **argv)
376 {
377     char    *p;
378     char    *pl;
379     char    *parg;
380     int     c;
381     int     i;
382     int     pgerrflg = 0; /* err flg: non-numeric arg w/p & g options */
383     size_t  size, len;
384     DIR     *dirp;
385     struct dirent *dentp;
386     pid_t   maxpid;
387     pid_t   id;
388     int     ret;
389     char    loc_stime_str[32];

391     (void) setlocale(LC_ALL, "");
392 #if !defined(TEXT_DOMAIN) /* Should be defined by cc -D */
393 #define TEXT_DOMAIN      "SYS_TEST" /* Use this only if it weren't */
394 #endif
395     (void) textdomain(TEXT_DOMAIN);

397     (void) memset(&euid_tbl, 0, sizeof (euid_tbl));
398     (void) memset(&ruid_tbl, 0, sizeof (ruid_tbl));
399     (void) memset(&egid_tbl, 0, sizeof (egid_tbl));
400     (void) memset(&rgid_tbl, 0, sizeof (rgid_tbl));

402     kbytes_per_page = sysconf(_SC_PAGESIZE) / 1024;

404     (void) gettimeofday(&now, NULL);

406     /*
407     * calculate width of pid fields based on configured MAXPID
408     * (must be at least 5 to retain output format compatibility)
409     */
410     id = maxpid = (pid_t)sysconf(_SC_MAXPID);
411     pidwidth = 1;
412     while ((id /= 10) > 0)
413         ++pidwidth;
414     pidwidth = pidwidth < 5 ? 5 : pidwidth;

416     fname[F_PID].width = fname[F_PPID].width = pidwidth;
417     fname[F_PGID].width = fname[F_SID].width = pidwidth;

419     /*
420     * TRANSLATION_NOTE
421     * Specify the printf format with width and precision for
422     * the STIME field.
423     */
424     len = snprintf(loc_stime_str, sizeof (loc_stime_str),
425                   dcgettext(NULL, "%8.8s", LC_TIME), "STIME");
426     if (len >= sizeof (loc_stime_str))
427         len = sizeof (loc_stime_str) - 1;

429     fname[F_STIME].width = fname[F_STIME].minwidth = len;

431     while ((c = getopt(argc, argv, "jlfceAadLPyZHh:t:p:g:u:U:G:n:s:o:z:"))
432           != EOF)
433         switch (c) {
434             case 'H':           /* Show home lgroups */

```

```

435         Hflg++;
436         break;
437     case 'h':
438         /*
439         * Show processes/threads with given home lgroups
440         */
441         hflg++;
442         pl = optarg;
443         do {
444             int id;

446             /*
447             * Get all IDs in the list, verify for
448             * correctness and place in lgrps array.
449             */
450             parg = getarg(&pl);
451             /* Convert string to integer */
452             ret = str2id(parg, (pid_t *)&id, 0,
453                       MAX_LGRP_ID);
454             /* Complain if ID didn't parse correctly */
455             if (ret != 0) {
456                 pgerrflg++;
457                 (void) fprintf(stderr,
458                               gettext("ps: %s "), parg);
459                 if (ret == EINVAL)
460                     (void) fprintf(stderr,
461                                   gettext("is an invalid "
462                                           "non-numeric argument"));
463                 else
464                     (void) fprintf(stderr,
465                                   gettext("exceeds valid "
466                                           "range"));
467                 (void) fprintf(stderr,
468                               gettext(" for -h option\n"));
469                 continue;
470             }

472             /* Extend lgrps array if needed */
473             if (nlgrps == lgrps_size) {
474                 /* Double the size of the lgrps array */
475                 if (lgrps_size == 0)
476                     lgrps_size = SIZ;
477                 lgrps_size *= 2;
478                 lgrps = Realloc(lgrps,
479                                lgrps_size * sizeof (int));
480             }
481             /* place the id in the lgrps table */
482             lgrps[nlgrps++] = id;
483         } while (*pl);
484         break;
485     case 'l':           /* long listing */
486         lflg++;
487         break;
488     case 'f':           /* full listing */
489         fflg++;
490         break;
491     case 'j':
492         jflg++;
493         break;
494     case 'c':
495         /*
496         * Format output to reflect scheduler changes:
497         * high numbers for high priorities and don't
498         * print nice or p_cpu values. 'c' option only
499         * effective when used with 'l' or 'f' options.
500         */

```



```

501         cflg++;
502         break;
503     case 'A':          /* list every process */
504     case 'e':          /* (obsolete) list every process */
505         Aflg++;
506         tflg = Gflg = Uflg = uflg = pflg = gflg = sflg = 0;
507         zflg = hflg = 0;
508         break;
509     case 'a':
510         /*
511          * Same as 'e' except no session group leaders
512          * and no non-terminal processes.
513          */
514         aflg++;
515         break;
516     case 'd':          /* same as e except no session leaders */
517         dflg++;
518         break;
519     case 'L':          /* show lwps */
520         Lflg++;
521         break;
522     case 'P':          /* show bound processor */
523         Pflg++;
524         break;
525     case 'W':          /* truncate long names */
526         Wflg++;
527         break;
528     case 'y':          /* omit F & ADDR, report RSS & SZ in Kby */
529         yflg++;
530         break;
531     case 'n':          /* no longer needed; retain as no-op */
532         (void) fprintf(stderr,
533             gettext("ps: warning: -n option ignored\n"));
534         break;
535     case 't':          /* terminals */
536     #define TSZ
537         30
538         tflg++;
539         pl = optarg;
540         do {
541             char nambuf[TSZ+6];      /* for "/dev/" + '\0' */
542             struct stat64 s;
543             parg = getarg(&pl);
544             p = Realloc(NULL, TSZ+1); /* for '\0' */
545             /* zero the buffer before using it */
546             p[0] = '\0';
547             size = TSZ;
548             if (isdigit(*parg)) {
549                 (void) strcpy(p, "tty");
550                 size -= 3;
551             }
552             (void) strncat(p, parg, size);
553             if (ntty == ttysz) {
554                 if ((ttysz *= 2) == 0)
555                     ttysz = NTTY5;
556                 tty = Realloc(tty,
557                     (ttysz + 1) * sizeof (struct tty));
558             }
559             tty[ntty].tdev = PRNODDEV;
560             (void) strcpy(nambuf, "/dev/");
561             (void) strcat(nambuf, p);
562             if (stat64(nambuf, &s) == 0)
563                 tty[ntty].tdev = s.st_rdev;
564             tty[ntty++].tname = p;
565         } while (*pl);
566     case 'p':          /* proc ids */

```

```

567         pflg++;
568         pl = optarg;
569         do {
570             pid_t id;
571             parg = getarg(&pl);
572             if ((ret = str2id(parg, &id, 0, maxpid)) != 0) {
573                 pgerrflg++;
574                 (void) fprintf(stderr,
575                     gettext("ps: %s "), parg);
576                 if (ret == EINVAL)
577                     (void) fprintf(stderr,
578                         gettext("is an invalid "
579                             "non-numeric argument"));
580                 else
581                     (void) fprintf(stderr,
582                         gettext("exceeds valid "
583                             "range"));
584                 (void) fprintf(stderr,
585                     gettext(" for -p option\n"));
586                 continue;
587             }
588         }
589         if (npid == pidsz) {
590             if ((pidsz *= 2) == 0)
591                 pidsz = SIZ;
592             pid = Realloc(pid,
593                 pidsz * sizeof (pid_t));
594         }
595         pid[npid++] = id;
596     } while (*pl);
597     break;
598     case 's':          /* session */
599         sflg++;
600         pl = optarg;
601         do {
602             pid_t id;
603             parg = getarg(&pl);
604             if ((ret = str2id(parg, &id, 0, maxpid)) != 0) {
605                 pgerrflg++;
606                 (void) fprintf(stderr,
607                     gettext("ps: %s "), parg);
608                 if (ret == EINVAL)
609                     (void) fprintf(stderr,
610                         gettext("is an invalid "
611                             "non-numeric argument"));
612                 else
613                     (void) fprintf(stderr,
614                         gettext("exceeds valid "
615                             "range"));
616                 (void) fprintf(stderr,
617                     gettext(" for -s option\n"));
618                 continue;
619             }
620         }
621         if (nssid == sessidsz) {
622             if ((sessidsz *= 2) == 0)
623                 sessidsz = SIZ;
624             sessid = Realloc(sessid,
625                 sessidsz * sizeof (pid_t));
626         }
627         sessid[nssid++] = id;
628     } while (*pl);
629     break;
630     case 'g':          /* proc group */

```

```

633     gflg++;
634     pl = optarg;
635     do {
636         pid_t id;
637
638         parg = getarg(&pl);
639         if ((ret = str2id(parg, &id, 0, maxpid)) != 0) {
640             pgerrflg++;
641             (void) fprintf(stderr,
642                 gettext("ps: %s "), parg);
643             if (ret == EINVAL)
644                 (void) fprintf(stderr,
645                     gettext("is an invalid "
646                         "non-numeric argument"));
647             else
648                 (void) fprintf(stderr,
649                     gettext("exceeds valid "
650                         "range"));
651             (void) fprintf(stderr,
652                 gettext(" for -g option\n"));
653             continue;
654         }
655
656         if (ngrpid == grpidsz) {
657             if ((grpidsz *= 2) == 0)
658                 grpidsz = SIZ;
659             grpidsz = SIZ;
660             grpidsz = Realloc(grpidsz * sizeof(pid_t));
661             grpidsz = SIZ;
662             grpidsz * sizeof(pid_t);
663             grpidsz * sizeof(pid_t);
664             } while (*pl);
665             break;
666         case 'u': /* effective user name or number */
667             uflg++;
668             pl = optarg;
669             do {
670                 parg = getarg(&pl);
671                 add_uentry(&uid_tbl, parg);
672             } while (*pl);
673             break;
674         case 'U': /* real user name or number */
675             Uflg++;
676             pl = optarg;
677             do {
678                 parg = getarg(&pl);
679                 add_uentry(&uid_tbl, parg);
680             } while (*pl);
681             break;
682         case 'G': /* real group name or number */
683             Gflg++;
684             pl = optarg;
685             do {
686                 parg = getarg(&pl);
687                 add_gentry(&gid_tbl, parg);
688             } while (*pl);
689             break;
690         case 'o': /* output format */
691             p = optarg;
692             while ((p = parse_format(p)) != NULL)
693                 ;
694             break;
695         case 'z': /* zone name or number */
696             zflg++;
697             pl = optarg;
698             do {
699                 zoneid_t id;

```

```

700         parg = getarg(&pl);
701         if (zone_get_id(parg, &id) != 0) {
702             pgerrflg++;
703             (void) fprintf(stderr,
704                 gettext("ps: unknown zone %s\n"),
705                 parg);
706             continue;
707         }
708
709         if (nzoneid == zoneidsz) {
710             if ((zoneidsz *= 2) == 0)
711                 zoneidsz = SIZ;
712             zoneidsz = SIZ;
713             zoneidsz = Realloc(zoneidsz * sizeof(zoneid_t));
714             zoneidsz * sizeof(zoneid_t);
715             zoneidsz * sizeof(zoneid_t);
716             zoneid[nzoneid++] = id;
717         } while (*pl);
718         break;
719         case 'Z': /* show zone name */
720             Zflg++;
721             break;
722         default: /* error on ? */
723             errflg++;
724             break;
725     }
726
727     if (errflg || optind < argc || pgerrflg)
728         usage();
729
730     if (tflg)
731         tty[ntty].tname = NULL;
732
733     /*
734     * If an appropriate option has not been specified, use the
735     * current terminal and effective uid as the default.
736     */
737     if (!(aflg|Aflg|dflg|Gflg|hflg|Uflg|uflg|tflg|pflg|gflg|sflg|zflg)) {
738         psinfo_t info;
739         int procfid;
740         char *name;
741         char pname[100];
742
743         /* get our own controlling tty name using /proc */
744         (void) snprintf(pname, sizeof(pname),
745             "%s/self/psinfo", procfid);
746         if ((procfid = open(pname, O_RDONLY)) < 0 ||
747             read(procfid, (char *)&info, sizeof(info)) < 0 ||
748             info.pr_ttydev == PRNODEV) {
749             (void) fprintf(stderr,
750                 gettext("ps: no controlling terminal\n"));
751             exit(1);
752         }
753         (void) close(procfid);
754
755         i = 0;
756         name = gettty(&info);
757         if (*name == '?') {
758             (void) fprintf(stderr,
759                 gettext("ps: can't find controlling terminal\n"));
760             exit(1);
761         }
762         if (ntty == ttysz) {
763             if ((ttysz *= 2) == 0)
764                 ttysz = NTTY;
765             ttysz = NTTY;
766             tty = Realloc(tty, (ttysz + 1) * sizeof(struct tty));
767         }

```

```

765         tty[ntty].tdev = info.pr_ttydev;
766         tty[ntty+].tname = name;
767         tty[ntty].tname = NULL;
768         tflg++;
769         tuid = getuid();
770     }
771     if (Aflg) {
772         Gflg = Uflg = uflg = pflg = sflg = gflg = aflg = dflg = 0;
773         zflg = hflg = 0;
774     }
775     if (Aflg | aflg | dflg)
776         tflg = 0;

778     i = 0;          /* prepare to exit on name lookup errors */
779     i += uconv(&euid_ttbl);
780     i += uconv(&ruid_ttbl);
781     i += gconv(&egid_ttbl);
782     i += gconv(&rgid_ttbl);
783     if (i)
784         exit(1);

786     /* allocate a buffer for lwpsinfo structures */
787     lpbufsize = 4096;
788     if (Lflg && (lpsinfobuf = malloc(lpbufsize)) == NULL) {
789         (void) fprintf(stderr,
790             gettext("ps: no memory\n"));
791         exit(1);
792     }

794     if (fields) { /* print user-specified header */
795         if (do_header) {
796             struct field *f;

798             for (f = fields; f != NULL; f = f->next) {
799                 if (f != fields)
800                     (void) printf(" ");
801                 switch (f->fname) {
802                     case F_TTY:
803                         (void) printf("%-*s",
804                             f->width, f->header);
805                         break;
806                     case F_FNAME:
807                     case F_COMM:
808                     case F_ARGS:
809                         /*
810                          * Print these headers full width
811                          * unless they appear at the end.
812                          */
813                         if (f->next != NULL) {
814                             (void) printf("%-*s",
815                                 f->width, f->header);
816                         } else {
817                             (void) printf("%s",
818                                 f->header);
819                         }
820                         break;
821                     default:
822                         (void) printf("%*s",
823                             f->width, f->header);
824                         break;
825                 }
826             }
827             (void) printf("\n");
828         }
829     } else { /* print standard header */
830

```

```

831         * All fields before 'PID' are printed with a trailing space
832         * as a separator and that is how we print the headers too.
833         */
834         if (lflg) {
835             if (yflg)
836                 (void) printf("S ");
837             else
838                 (void) printf(" F S ");
839         }
840         if (Zflg)
841             (void) printf("    ZONE ");
842         if (fflg) {
843             (void) printf("    UID ");
844         } else if (lflg)
845             (void) printf("    UID ");

847         (void) printf("%*s", pidwidth, "PID");
848         if (lflg || fflg)
849             (void) printf(" %*s", pidwidth, "PPID");
850         if (jflg)
851             (void) printf(" %*s %*s", pidwidth, "PGID",
852                 pidwidth, "SID");
853         if (Lflg)
854             (void) printf("    LWP");
855         if (Pflg)
856             (void) printf("    PSR");
857         if (Lflg && fflg)
858             (void) printf("    NLWP");
859         if (cflg)
860             (void) printf("    CLS PRI");
861         else if (lflg || fflg) {
862             (void) printf("    C");
863             if (lflg)
864                 (void) printf(" PRI NI");
865         }
866         if (lflg) {
867             if (yflg)
868                 (void) printf("    RSS    SZ    WCHAN");
869             else
870                 (void) printf("    ADDR    SZ    WCHAN");
871         }
872         if (fflg)
873             (void) printf(" %s", loc_stime_str);
874         if (Hflg)
875             (void) printf("    LGRP");
876         if (Lflg)
877             (void) printf("    TTY        LTIME CMD\n");
878         else
879             (void) printf("    TTY        TIME CMD\n");
880     }

883     if (pflg && !(aflg|Aflg|dflg|Gflg|Uflg|uflg|hflg|tflg|gflg|sflg|zflg) &&
884         npid <= PTHRESHOLD) {
885         /*
886          * If we are looking at specific processes go straight
887          * to their /proc entries and don't scan /proc.
888          */
889         int i;

891         (void) qsort(pid, npid, sizeof(pid_t), pidcmp);
892         for (i = 0; i < npid; i++) {
893             char pname[12];

895             if (i >= 1 && pid[i] == pid[i - 1])
896                 continue;

```

```

897         (void) sprintf(pname, "%d", (int)pid[i]);
898         if (print_proc(pname) == 0)
899             retcode = 0;
900     }
901 } else {
902     /*
903     * Determine which processes to print info about by searching
904     * the /proc directory and looking at each process.
905     */
906     if ((dirp = opendir(procdir)) == NULL) {
907         (void) fprintf(stderr,
908             gettext("ps: cannot open PROC directory %s\n"),
909             procdir);
910         exit(1);
911     }
912
913     /* for each active process --- */
914     while (dentp = readdir(dirp)) {
915         if (dentp->d_name[0] == '.') /* skip . and .. */
916             continue;
917         if (print_proc(dentp->d_name) == 0)
918             retcode = 0;
919     }
920
921     (void) closedir(dirp);
922 }
923 return (retcode);
924 }

```

unchanged\_portion\_omitted\_

```

1060 static void
1061 usage(void) /* print usage message and quit */
1062 {
1063     struct def_field *df, *sorted[NFIELDS];
1064     int pos = 80, i = 0;
1065
1066     static char usage1[] =
1067         "ps [ -aAdefHlcljLPwYZ ] [ -o format ] [ -t termlist ]";
1068     static char usage2[] =
1069         "ps [ -aAdefHlcljLPwYZ ] [ -o format ] [ -t termlist ]";
1070     static char usage3[] =
1071         "\t[ -u userlist ] [ -U userlist ] [ -G grouplist ]";
1072     static char usage4[] =
1073         "\t[ -p proclis ] [ -g pgrplist ] [ -s sidlist ]";
1074     static char usage5[] =
1075         "\t[ -z zonelist ] [ -h lgrplist ]";
1076     static char usage6[] =
1077         " 'format' is one or more of:";
1078
1079     (void) fprintf(stderr,
1080         gettext("usage: %s\n%s\n%s\n%s\n%s\n"),
1081         gettext(usage1), gettext(usage2), gettext(usage3),
1082         gettext(usage4), gettext(usage5));
1083
1084     /*
1085     * Now print out the possible output formats such that they neatly fit
1086     * into eighty columns. Note that the fact that we are determining
1087     * this output programmatically means that a gettext() is impossible --
1088     * but it would be a mistake to localize the output formats anyway as
1089     * they are tokens for input, not output themselves.
1090     */
1091     for (df = &fname[0]; df < &fname[NFIELDS]; df++)
1092         sorted[i++] = df;
1093
1094     (void) qsort(sorted, NFIELDS, sizeof (void *), field_cmp);
1095
1096     for (i = 0; i < NFIELDS; i++) {

```

```

1095         if (pos + strlen((df = sorted[i])>fname) + 1 >= 80) {
1096             (void) fprintf(stderr, "\n\t");
1097             pos = 8;
1098         }
1099
1100         (void) fprintf(stderr, "%s%s", pos > 8 ? " " : "", df->fname);
1101         pos += strlen(df->fname) + 1;
1102     }
1103
1104     (void) fprintf(stderr, "\n");
1105
1106     exit(1);
1107 }

```

unchanged\_portion\_omitted\_

```

1355 /*
1356 * Print info about the process.
1357 */
1358 static void
1359 prcom(psinfo_t *psinfo, char *tty)
1360 {
1361     char *cp;
1362     long tm;
1363     int bytesleft;
1364     int wcnt, length;
1365     wchar_t wchar;
1366     struct passwd *pwd;
1367     int zombie_lwp;
1368     char zonename[ZONENAME_MAX];
1369
1370     /*
1371     * If process is zombie, call zombie print routine and return.
1372     */
1373     if (psinfo->pr_nlwp == 0) {
1374         if (fields != NULL)
1375             pr_fields(psinfo, tty, print_zombie_field);
1376         else
1377             przom(psinfo);
1378         return;
1379     }
1380
1381     zombie_lwp = (Lflg && psinfo->pr_lwp.pr_sname == 'Z');
1382
1383     /*
1384     * If user specified '-o format', print requested fields and return.
1385     */
1386     if (fields != NULL) {
1387         pr_fields(psinfo, tty, print_field);
1388         return;
1389     }
1390
1391     /*
1392     * All fields before 'PID' are printed with a trailing space as a
1393     * separator, rather than keeping track of which column is first. All
1394     * other fields are printed with a leading space.
1395     */
1396     if (lflg) {
1397         if (!yflg)
1398             (void) printf("%2x ", psinfo->pr_flag & 0377); /* F */
1399         (void) printf("%c ", psinfo->pr_lwp.pr_sname); /* S */
1400     }
1401
1402     if (Zflg) { /* ZONE */
1403         if (getzonenamebyid(psinfo->pr_zoneid, zonename,
1404             sizeof (zonename)) < 0) {
1405             if (snprintf(NULL, 0, "%d",

```

```

1406         ((int)psinfo->pr_zoneid) > 7)
1407         (void) printf(" %6.6d%c ",
1408         ((int)psinfo->pr_zoneid), '**');
1409     else
1410         (void) printf(" %7.7d ",
1411         ((int)psinfo->pr_zoneid));
1398     (void) printf(" %7.7d ", ((int)psinfo->pr_zoneid));
1412 } else {
1413     if (strlen(zonename) > 8)
1414         (void) printf("%7.7s%c ", zonename, '**');
1415     else
1416         (void) printf("%8.8s ", zonename);
1417 }
1418 }
1420 if (fflg) {
1421     if ((pwd = getpwuid(psinfo->pr_euid)) != NULL) { /* UID */
1422         if (strlen(pwd->pw_name) > 8)
1423             (void) printf("%7.7s%c ", pwd->pw_name, '**');
1424         else
1425             (void) printf("%8.8s ", pwd->pw_name);
1426     } else {
1427         if (snprintf(NULL, 0, "%u",
1428         ((int)psinfo->pr_euid)) > 7)
1429             (void) printf(" %6.6u%c ", psinfo->pr_euid,
1430             '**');
1431         else
1432             (void) printf(" %7.7u ", psinfo->pr_euid);
1433     }
1434 } else if (lflg) {
1435     if (snprintf(NULL, 0, "%u", ((int)psinfo->pr_euid)) > 6)
1436         (void) printf(" %5.5u%c ", psinfo->pr_euid, '**');
1437     else
1438         (void) printf(" %6u ", psinfo->pr_euid);
1439 }
1440 (void) printf("%*d", pidwidth, (int)psinfo->pr_pid); /* PID */
1441 if (lflg || fflg)
1442     (void) printf(" %*d", pidwidth,
1443     (int)psinfo->pr_ppid); /* PPID */
1444 if (jflg) {
1445     (void) printf(" %*d", pidwidth,
1446     (int)psinfo->pr_pgid); /* PGID */
1447     (void) printf(" %*d", pidwidth,
1448     (int)psinfo->pr_sid); /* SID */
1449 }
1450 if (Lflg)
1451     (void) printf(" %5d", (int)psinfo->pr_lwp.pr_lwpid); /* LWP */
1452 if (Pflg) {
1453     if (psinfo->pr_lwp.pr_bindpro == PBIND_NONE) /* PSR */
1454         (void) printf(" -");
1455     else
1456         (void) printf(" %3d", psinfo->pr_lwp.pr_bindpro);
1457 }
1458 if (Lflg && fflg) /* NLWP */
1459     (void) printf(" %5d", psinfo->pr_nlwp + psinfo->pr_nzomb);
1460 if (cflg) {
1461     if (zombie_lwp) /* CLS */
1462         (void) printf(" ");
1463     else
1464         (void) printf(" %4s", psinfo->pr_lwp.pr_clname);
1465     (void) printf(" %3d", psinfo->pr_lwp.pr_pri); /* PRI */
1466 } else if (lflg || fflg) {
1467     (void) printf(" %3d", psinfo->pr_lwp.pr_cpu & 0377); /* C */
1468     if (lflg) { /* PRI NI */
1469         /*

```

```

1470     * Print priorities the old way (lower numbers
1471     * mean higher priority) and print nice value
1472     * for time sharing procs.
1473     */
1474     (void) printf(" %3d", psinfo->pr_lwp.pr_oldpri);
1475     if (psinfo->pr_lwp.pr_oldpri != 0)
1476         (void) printf(" %2d", psinfo->pr_lwp.pr_nice);
1477     else
1478         (void) printf(" %2.2s",
1479         psinfo->pr_lwp.pr_clname);
1480 }
1481 }
1482 if (lflg) {
1483     if (yflg) {
1484         if (psinfo->pr_flag & SSYS) /* RSS */
1485             (void) printf(" 0");
1486         else if (psinfo->pr_rssize)
1487             (void) printf(" %5lu",
1488             (ulong_t)psinfo->pr_rssize);
1489         else
1490             (void) printf(" ?");
1491         if (psinfo->pr_flag & SSYS) /* SZ */
1492             (void) printf(" 0");
1493         else if (psinfo->pr_size)
1494             (void) printf(" %6lu",
1495             (ulong_t)psinfo->pr_size);
1496         else
1497             (void) printf(" ?");
1498     } else {
1499 #ifndef _LP64
1500         if (psinfo->pr_addr) /* ADDR */
1501             (void) printf(" %8lx",
1502             (ulong_t)psinfo->pr_addr);
1503         else
1504             (void) printf(" ?");
1505 #endif
1506         if (psinfo->pr_flag & SSYS) /* SZ */
1507             (void) printf(" 0");
1508         else if (psinfo->pr_size)
1509             (void) printf(" %6lu",
1510             (ulong_t)psinfo->pr_size / kbytes_per_page);
1511         else
1512             (void) printf(" ?");
1513     }
1514     if (psinfo->pr_lwp.pr_sname != 'S') /* WCHAN */
1515         (void) printf(" ");
1516 #ifndef _LP64
1517     else if (psinfo->pr_lwp.pr_wchan)
1518         (void) printf(" %8lx",
1519         (ulong_t)psinfo->pr_lwp.pr_wchan);
1520 #endif
1521     else
1522         (void) printf(" ?");
1523 }
1524 if (fflg) { /* STIME */
1525     int width = fname[F_STIME].width;
1526     if (Lflg)
1527         prtime(psinfo->pr_lwp.pr_start, width + 1, 1);
1528     else
1529         prtime(psinfo->pr_start, width + 1, 1);
1530 }
1531 if (Hflg) {
1532     /* Display home lgroup */
1533     (void) printf(" %4d", (int)psinfo->pr_lwp.pr_lgrp);
1534 }
1535 }

```

```

1537     (void) printf(" %-8.14s", ttyp);          /* TTY */
1538     if (Lflg) {
1539         tm = psinfo->pr_lwp.pr_time.tv_sec;
1540         if (psinfo->pr_lwp.pr_time.tv_nsec > 500000000)
1541             tm++;
1542     } else {
1543         tm = psinfo->pr_time.tv_sec;
1544         if (psinfo->pr_time.tv_nsec > 500000000)
1545             tm++;
1546     }
1547     (void) printf(" %4ld:%.2ld", tm / 60, tm % 60);    /* [L]TIME */

1549     if (zombie_lwp) {
1550         (void) printf(" <defunct>\n");
1551         return;
1552     }

1554     if (!fflg) {                                /* CMD */
1555         wcnt = namencnt(psinfo->pr_fname, 16, 8);
1556         (void) printf(" %.*s\n", wcnt, psinfo->pr_fname);
1557         return;
1558     }

1561     /*
1562     * PRARGSZ == length of cmd arg string.
1563     */
1564     psinfo->pr_psargs[PRARGSZ-1] = '\0';
1565     bytesleft = PRARGSZ;
1566     for (cp = psinfo->pr_psargs; *cp != '\0'; cp += length) {
1567         length = mbtowc(&wchar, cp, MB_LEN_MAX);
1568         if (length == 0)
1569             break;
1570         if (length < 0 || !iswprint(wchar)) {
1571             if (length < 0)
1572                 length = 1;
1573             if (bytesleft <= length) {
1574                 *cp = '\0';
1575                 break;
1576             }
1577             /* omit the unprintable character */
1578             (void) memmove(cp, cp+length, bytesleft-length);
1579             length = 0;
1580         }
1581         bytesleft -= length;
1582     }
1583     wcnt = namencnt(psinfo->pr_psargs, PRARGSZ, lflg ? 35 : PRARGSZ);
1584     (void) printf(" %.*s\n", wcnt, psinfo->pr_psargs);
1585 }

```

unchanged portion omitted

```

1639 static void
1640 print_field(psinfo_t *psinfo, struct field *f, const char *ttyp)
1641 {
1642     int width = f->width;
1643     struct passwd *pwd;
1644     struct group *grp;
1645     time_t cputime;
1646     int bytesleft;
1647     int wcnt;
1648     wchar_t wchar;
1649     char *cp;
1650     int length;
1651     ulong_t mask;
1652     char c, *csave;

```

```

1653     int zombie_lwp;

1655     zombie_lwp = (Lflg && psinfo->pr_lwp.pr_sname == 'Z');

1657     switch (f->fname) {
1658     case F_RUSER:
1659         if ((pwd = getpwuid(psinfo->pr_uid)) != NULL) {
1660             if (Wflg && strlen(pwd->pw_name) > width)
1661                 (void) printf("%.*s%c", width - 1,
1662                     pwd->pw_name, '*');
1663             else
1664                 if ((pwd = getpwuid(psinfo->pr_uid)) != NULL)
1665                     (void) printf("%*s", width, pwd->pw_name);
1666         } else {
1667             if (Wflg && snprintf(NULL, 0, "%u",
1668                 ((int)psinfo->pr_uid)) > width)
1669                 (void) printf("%*u%c", width - 1,
1670                     psinfo->pr_uid, '*');
1671             else
1672                 (void) printf("%*u", width, psinfo->pr_uid);
1673         }
1674         break;
1675     case F_USER:
1676         if ((pwd = getpwuid(psinfo->pr_euid)) != NULL) {
1677             if (Wflg && strlen(pwd->pw_name) > width)
1678                 (void) printf("%.*s%c", width - 1,
1679                     pwd->pw_name, '*');
1680             else
1681                 if ((pwd = getpwuid(psinfo->pr_euid)) != NULL)
1682                     (void) printf("%*s", width, pwd->pw_name);
1683         } else {
1684             if (Wflg && snprintf(NULL, 0, "%u",
1685                 ((int)psinfo->pr_euid)) > width)
1686                 (void) printf("%*u%c", width - 1,
1687                     psinfo->pr_euid, '*');
1688             else
1689                 (void) printf("%*u", width, psinfo->pr_euid);
1690         }
1691         break;
1692     case F_RGROUP:
1693         if ((grp = getgrgid(psinfo->pr_gid)) != NULL)
1694             (void) printf("%*s", width, grp->gr_name);
1695         else
1696             (void) printf("%*u", width, psinfo->pr_gid);
1697         break;
1698     case F_GROUP:
1699         if ((grp = getgrgid(psinfo->pr_egid)) != NULL)
1700             (void) printf("%*s", width, grp->gr_name);
1701         else
1702             (void) printf("%*u", width, psinfo->pr_egid);
1703         break;
1704     case F_RUID:
1705         (void) printf("%*u", width, psinfo->pr_uid);
1706         break;
1707     case F_UID:
1708         (void) printf("%*u", width, psinfo->pr_euid);
1709         break;
1710     case F_RGID:
1711         (void) printf("%*u", width, psinfo->pr_gid);
1712         break;
1713     case F_GID:
1714         (void) printf("%*u", width, psinfo->pr_egid);
1715         break;
1716     case F_PID:

```

```

1717         (void) printf("%*d", width, (int)psinfo->pr_pid);
1718         break;
1719     case F_PPID:
1720         (void) printf("%*d", width, (int)psinfo->pr_ppid);
1721         break;
1722     case F_PGID:
1723         (void) printf("%*d", width, (int)psinfo->pr_pgid);
1724         break;
1725     case F_SID:
1726         (void) printf("%*d", width, (int)psinfo->pr_sid);
1727         break;
1728     case F_PSR:
1729         if (zombie_lwp || psinfo->pr_lwp.pr_bindpro == PBIND_NONE)
1730             (void) printf("%*s", width, "-");
1731         else
1732             (void) printf("%*d", width, psinfo->pr_lwp.pr_bindpro);
1733         break;
1734     case F_LWP:
1735         (void) printf("%*d", width, (int)psinfo->pr_lwp.pr_lwpid);
1736         break;
1737     case F_NLWP:
1738         (void) printf("%*d", width, psinfo->pr_nlwp + psinfo->pr_nzomb);
1739         break;
1740     case F_OPRI:
1741         if (zombie_lwp)
1742             (void) printf("%*s", width, "-");
1743         else
1744             (void) printf("%*d", width, psinfo->pr_lwp.pr_oldpri);
1745         break;
1746     case F_PRI:
1747         if (zombie_lwp)
1748             (void) printf("%*s", width, "-");
1749         else
1750             (void) printf("%*d", width, psinfo->pr_lwp.pr_pri);
1751         break;
1752     case F_F:
1753         mask = 0xffffffffUL;
1754         if (width < 8)
1755             mask >>= (8 - width) * 4;
1756         (void) printf("%*lx", width, psinfo->pr_flag & mask);
1757         break;
1758     case F_S:
1759         (void) printf("%*c", width, psinfo->pr_lwp.pr_sname);
1760         break;
1761     case F_C:
1762         if (zombie_lwp)
1763             (void) printf("%*s", width, "-");
1764         else
1765             (void) printf("%*d", width, psinfo->pr_lwp.pr_cpu);
1766         break;
1767     case F_PCPU:
1768         if (zombie_lwp)
1769             (void) printf("%*s", width, "-");
1770         else if (Lflg)
1771             prtpct(psinfo->pr_lwp.pr_pctcpu, width);
1772         else
1773             prtpct(psinfo->pr_pctcpu, width);
1774         break;
1775     case F_PMEM:
1776         prtpct(psinfo->pr_pctmem, width);
1777         break;
1778     case F_OSZ:
1779         (void) printf("%*lu", width,
1780             (ulong_t)psinfo->pr_size / kbytes_per_page);
1781         break;
1782     case F_VSZ:

```

```

1783         (void) printf("%*lu", width, (ulong_t)psinfo->pr_size);
1784         break;
1785     case F_RSS:
1786         (void) printf("%*lu", width, (ulong_t)psinfo->pr_rssize);
1787         break;
1788     case F_NICE:
1789         /* if pr_oldpri is zero, then this class has no nice */
1790         if (zombie_lwp)
1791             (void) printf("%*s", width, "-");
1792         else if (psinfo->pr_lwp.pr_oldpri != 0)
1793             (void) printf("%*d", width, psinfo->pr_lwp.pr_nice);
1794         else
1795             (void) printf("%*.s", width, width,
1796                 psinfo->pr_lwp.pr_clname);
1797         break;
1798     case F_CLASS:
1799         if (zombie_lwp)
1800             (void) printf("%*s", width, "-");
1801         else
1802             (void) printf("%*.s", width, width,
1803                 psinfo->pr_lwp.pr_clname);
1804         break;
1805     case F_STIME:
1806         if (Lflg)
1807             prtime(psinfo->pr_lwp.pr_start, width, 0);
1808         else
1809             prtime(psinfo->pr_start, width, 0);
1810         break;
1811     case F_ETIME:
1812         if (Lflg)
1813             print_time(delta_secs(&psinfo->pr_lwp.pr_start),
1814                 width);
1815         else
1816             print_time(delta_secs(&psinfo->pr_start), width);
1817         break;
1818     case F_TIME:
1819         if (Lflg) {
1820             cputime = psinfo->pr_lwp.pr_time.tv_sec;
1821             if (psinfo->pr_lwp.pr_time.tv_nsec > 500000000)
1822                 cputime++;
1823         } else {
1824             cputime = psinfo->pr_time.tv_sec;
1825             if (psinfo->pr_time.tv_nsec > 500000000)
1826                 cputime++;
1827         }
1828         print_time(cputime, width);
1829         break;
1830     case F_TTY:
1831         (void) printf("%*s", width, tty);
1832         break;
1833     case F_ADDR:
1834         if (zombie_lwp)
1835             (void) printf("%*s", width, "-");
1836         else if (Lflg)
1837             (void) printf("%*lx", width,
1838                 (long)psinfo->pr_lwp.pr_addr);
1839         else
1840             (void) printf("%*lx", width, (long)psinfo->pr_addr);
1841         break;
1842     case F_WCHAN:
1843         if (!zombie_lwp && psinfo->pr_lwp.pr_wchan)
1844             (void) printf("%*lx", width,
1845                 (long)psinfo->pr_lwp.pr_wchan);
1846         else
1847             (void) printf("%*.s", width, width, "-");
1848         break;

```

```

1849     case F_FNAME:
1850         /*
1851          * Print full width unless this is the last output format.
1852          */
1853         if (zombie_lwp) {
1854             if (f->next != NULL)
1855                 (void) printf("%-*s", width, "<defunct>");
1856             else
1857                 (void) printf("%s", "<defunct>");
1858             break;
1859         }
1860         wcnt = namencnt(psinfo->pr_fname, 16, width);
1861         if (f->next != NULL)
1862             (void) printf("%-*.*s", width, wcnt, psinfo->pr_fname);
1863         else
1864             (void) printf("%-.*s", wcnt, psinfo->pr_fname);
1865         break;
1866     case F_COMM:
1867         if (zombie_lwp) {
1868             if (f->next != NULL)
1869                 (void) printf("%-*s", width, "<defunct>");
1870             else
1871                 (void) printf("%s", "<defunct>");
1872             break;
1873         }
1874         csave = strpbrk(psinfo->pr_psargs, " \\t\\r\\v\\f\\n");
1875         if (csave) {
1876             c = *csave;
1877             *csave = '\\0';
1878         }
1879         /* FALLTHROUGH */
1880     case F_ARGS:
1881         /*
1882          * PRARGSZ == length of cmd arg string.
1883          */
1884         if (zombie_lwp) {
1885             (void) printf("%-*s", width, "<defunct>");
1886             break;
1887         }
1888         psinfo->pr_psargs[PRARGSZ-1] = '\\0';
1889         bytesleft = PRARGSZ;
1890         for (cp = psinfo->pr_psargs; *cp != '\\0'; cp += length) {
1891             length = mbtowc(&wchar, cp, MB_LEN_MAX);
1892             if (length == 0)
1893                 break;
1894             if (length < 0 || !iswprint(wchar)) {
1895                 if (length < 0)
1896                     length = 1;
1897                 if (bytesleft <= length) {
1898                     *cp = '\\0';
1899                     break;
1900                 }
1901                 /* omit the unprintable character */
1902                 (void) memmove(cp, cp+length, bytesleft-length);
1903                 length = 0;
1904             }
1905             bytesleft -= length;
1906         }
1907         wcnt = namencnt(psinfo->pr_psargs, PRARGSZ, width);
1908         /*
1909          * Print full width unless this is the last format.
1910          */
1911         if (f->next != NULL)
1912             (void) printf("%-*.*s", width, wcnt,
1913                 psinfo->pr_psargs);
1914         else

```

```

1915         (void) printf("%-.*s", wcnt,
1916             psinfo->pr_psargs);
1917         if (f->fname == F_COMM && csave)
1918             *csave = c;
1919         break;
1920     case F_TASKID:
1921         (void) printf("%*d", width, (int)psinfo->pr_taskid);
1922         break;
1923     case F_PROJID:
1924         (void) printf("%*d", width, (int)psinfo->pr_projid);
1925         break;
1926     case F_PROJECT:
1927         {
1928             struct project cproj;
1929             char proj_buf[PROJECT_BUFSZ];
1930
1931             if ((getprojbyid(psinfo->pr_projid, &cproj,
1932                 (void *)&proj_buf, PROJECT_BUFSZ)) == NULL) {
1933                 if (Wflg && snprintf(NULL, 0, "%d",
1934                     ((int)psinfo->pr_projid) > width)
1935                     (void) printf("%.*d%c", width - 1,
1936                         ((int)psinfo->pr_projid), '*'));
1937                 else
1938                     (void *)&proj_buf, PROJECT_BUFSZ)) == NULL)
1939                     (void) printf("%*d", width,
1940                         (int)psinfo->pr_projid);
1941             } else {
1942                 if (Wflg && cproj.pj_name != NULL &&
1943                     strlen(cproj.pj_name) > width)
1944                     (void) printf("%.*s%c", width - 1,
1945                         cproj.pj_name, '*');
1946                 else
1947                     (void) printf("%*s", width,
1948                         (cproj.pj_name != NULL) ?
1949                         cproj.pj_name : "---");
1950             }
1951             break;
1952     case F_PSET:
1953         if (zombie_lwp || psinfo->pr_lwp.pr_bindpset == PS_NONE)
1954             (void) printf("%*s", width, "-");
1955         else
1956             (void) printf("%*d", width, psinfo->pr_lwp.pr_bindpset);
1957         break;
1958     case F_ZONEID:
1959         (void) printf("%*d", width, (int)psinfo->pr_zoneid);
1960         break;
1961     case F_ZONE:
1962         {
1963             char zonename[ZONENAME_MAX];
1964
1965             if (getzonenamebyid(psinfo->pr_zoneid, zonename,
1966                 sizeof (zonename)) < 0) {
1967                 if (Wflg && snprintf(NULL, 0, "%d",
1968                     ((int)psinfo->pr_zoneid) > width)
1969                     (void) printf("%.*d%c", width - 1,
1970                         ((int)psinfo->pr_zoneid), '*'));
1971                 else
1972                     (void) printf("%*d", width,
1973                         (int)psinfo->pr_zoneid);
1974             } else {
1975                 ((int)psinfo->pr_zoneid);
1976                 if (Wflg && strlen(zonename) > width)
1977                     (void) printf("%.*s%c", width - 1,
1978                         zonename, '*');
1979                 else

```



```

1979         (void) printf("%*s", width, zonename);
1980     }
1981     }
1982     break;
1983 case F_CTID:
1984     if (psinfo->pr_contract == -1)
1985         (void) printf("%*s", width, "-");
1986     else
1987         (void) printf("%*ld", width, (long)psinfo->pr_contract);
1988     break;
1989 case F_LGRP:
1990     /* Display home lgroup */
1991     (void) printf("%*d", width, (int)psinfo->pr_lwp.pr_lgrp);
1992     break;
1993
1994 case F_DMODEL:
1995     (void) printf("%*s", width,
1996         psinfo->pr_dmodel == PR_MODEL_LP64 ? "_LP64" : "_ILP32");
1997     break;
1998 }
1999 }

```

unchanged portion omitted

```

2234 static void
2235 przom(psinfo_t *psinfo)
2236 {
2237     long tm;
2238     struct passwd *pwd;
2239     char zonename[ZONENAME_MAX];
2240
2241     /*
2242      * All fields before 'PID' are printed with a trailing space as a
2243      * spearator, rather than keeping track of which column is first. All
2244      * other fields are printed with a leading space.
2245      */
2246     if (lflg) { /* F S */
2247         if (!yflg)
2248             (void) printf("%2x ", psinfo->pr_flag & 0377); /* F */
2249         (void) printf("%c ", psinfo->pr_lwp.pr_sname); /* S */
2250     }
2251     if (Zflg) {
2252         if (getzonenamebyid(psinfo->pr_zoneid, zonename,
2253             sizeof (zonename)) < 0) {
2254             if (snprintf(NULL, 0, "%d",
2255                 ((int)psinfo->pr_zoneid) > 7)
2256                 (void) printf(" %6.6d%c ",
2257                     ((int)psinfo->pr_zoneid), '*');
2258             else
2259                 (void) printf(" %7.7d ",
2260                     ((int)psinfo->pr_zoneid));
2261         } else {
2262             if (strlen(zonename) > 8)
2263                 (void) printf(" %7.7s%c ", zonename, '*');
2264             else
2265                 (void) printf(" %8.8s ", zonename);
2266         }
2267     }
2268     if (Hflg) {
2269         /* Display home lgroup */
2270         (void) printf(" %6d", (int)psinfo->pr_lwp.pr_lgrp); /* LGRP */
2271     }
2272     if (fflg) {
2273         if ((pwd = getpwuid(psinfo->pr_euid)) != NULL) {
2274             if (strlen(pwd->pw_name) > 8)
2275                 (void) printf(" %7.7s%c ", pwd->pw_name, '*');

```

```

2276     else
2277         if ((pwd = getpwuid(psinfo->pr_euid)) != NULL)
2278             (void) printf(" %8.8s ", pwd->pw_name);
2279     } else {
2280         if (snprintf(NULL, 0, "%u",
2281             ((int)psinfo->pr_euid) > 7)
2282             (void) printf(" %6.6u%c ", psinfo->pr_euid,
2283                 '*');
2284         else
2285             (void) printf(" %7.7u ", psinfo->pr_euid);
2286     } else if (lflg) {
2287         if (snprintf(NULL, 0, "%u", ((int)psinfo->pr_euid) > 6)
2288             (void) printf(" %5.5u%c ", psinfo->pr_euid, '*');
2289         else
2290             (void) printf(" %6u ", psinfo->pr_euid);
2291     }
2292
2293     (void) printf("%*d", pidwidth, (int)psinfo->pr_pid); /* PID */
2294     if (lflg || fflg)
2295         (void) printf(" %*d", pidwidth,
2296             (int)psinfo->pr_ppid); /* PPID */
2297
2298     if (jflg) {
2299         (void) printf(" %*d", pidwidth,
2300             (int)psinfo->pr_pgid); /* PGID */
2301         (void) printf(" %*d", pidwidth,
2302             (int)psinfo->pr_sid); /* SID */
2303     }
2304
2305     if (Lflg)
2306         (void) printf(" %5d", 0); /* LWP */
2307     if (Pflg)
2308         (void) printf(" -"); /* PSR */
2309     if (Lflg && fflg)
2310         (void) printf(" %5d", 0); /* NLWP */
2311
2312     if (cflg) {
2313         (void) printf(" %4s", "-"); /* zombies have no class */
2314         (void) printf(" %3d", psinfo->pr_lwp.pr_pri); /* PRI */
2315     } else if (lflg || fflg) {
2316         (void) printf(" %3d", psinfo->pr_lwp.pr_cpu & 0377); /* C */
2317         if (lflg)
2318             (void) printf(" %3d %2s",
2319                 psinfo->pr_lwp.pr_oldpri, "-"); /* PRI NI */
2320     }
2321     if (lflg) {
2322         if (yflg) /* RSS SZ WCHAN */
2323             (void) printf(" %5d %6d %8s", 0, 0, "-");
2324         else /* ADDR SZ WCHAN */
2325             (void) printf(" %8s %6d %8s", "-", 0, "-");
2326     }
2327     if (fflg) {
2328         int width = fname[F_STIME].width;
2329         (void) printf(" %*.*s", width, width, "-"); /* STIME */
2330     }
2331     (void) printf(" %-8.14s", "?"); /* TTY */
2332
2333     tm = psinfo->pr_time.tv_sec;
2334     if (psinfo->pr_time.tv_nsec > 500000000)
2335         tm++;
2336     (void) printf(" %4ld:%.2ld", tm / 60, tm % 60); /* TIME */
2337     (void) printf(" <defunct>\n");
2338 }

```

unchanged portion omitted

```

*****
5355 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/pwck/pwck.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
24 * Copyright 2007 Sun Microsystems, Inc. All rights reserved.
25 * Use is subject to license terms.
26 */

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

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

32 #include <sys/types.h>
33 #include <sys/param.h>
34 #include <sys/signal.h>
35 #include <sys/sysmacros.h>
36 #include <sys/stat.h>
37 #include <stdio.h>
38 #include <stdlib.h>
39 #include <string.h>
40 #include <ctype.h>
41 #include <locale.h>
42 #include <errno.h>
43 #include <unistd.h>
44 #include <limits.h>

46 #define ERROR1      "Too many/few fields"
47 #define ERROR2      "Bad character(s) in logname"
48 #define ERROR2a     "First char in logname not alphabetic"
49 #define ERROR2b     "Logname field NULL"
50 #define ERROR2c     "Logname contains no lower-case letters"
51 #define ERROR3      "Logname too long/short"
52 #define ERROR4      "Invalid UID"
53 #define ERROR5      "Invalid GID"
54 #define ERROR6      "Login directory not found"
55 #define ERROR6a     "Login directory null"
56 #define ERROR7      "Optional shell file not found"

58 static int eflag, code = 0;

```

```

59 static int badc;
60 static int lc;
61 static char buf[512];
62 static void error(char *);

64 int
65 main(int argc, char **argv)
66 {
67     int delim[512];
68     char logbuf[512];
69     FILE *fptr;
70     struct stat obuf;
71     uid_t uid;
72     gid_t gid;
73     int i, j, colons;
74     char *pw_file;
75     struct stat stat_buf;
76     char *str, *lastc;

78     (void) setlocale(LC_ALL, "");

80 #if !defined(TEXT_DOMAIN)      /* Should be defined by cc -D */
81 #define TEXT_DOMAIN "SYS_TEST"
82 #endif
83     (void) textdomain(TEXT_DOMAIN);

85     if (argc == 1)
86         pw_file = "/etc/passwd";
87     else
88         pw_file = argv[1];

90     if ((fptr = fopen(pw_file, "r")) == NULL) {
91         (void) fprintf(stderr, gettext("cannot open %s\n"), pw_file);
92         exit(1);
93     }

95     if (fstat(fileno(fptr), &stat_buf) < 0) {
96         (void) fprintf(stderr, gettext("fstat failed for %s\n"),
97             pw_file);
98         (void) fclose(fptr);
99         exit(1);
100    }

102    if (stat_buf.st_size == 0) {
103        (void) fprintf(stderr, gettext("file %s is empty\n"), pw_file);
104        (void) fclose(fptr);
105        exit(1);
106    }

108    while (fgets(buf, sizeof(buf), fptr) != NULL) {

110        colons = 0;
111        badc = 0;
112        lc = 0;
113        eflag = 0;

115        /* Check that entry is not a nameservice redirection */

117        if (buf[0] == '+' || buf[0] == '-') {
118            /*
119             * Should set flag here to allow special case checking
120             * in the rest of the code,
121             * but for now, we'll just ignore this entry.
122             */
123            continue;
124        }

```

```

126         /* Check number of fields */
128     for (i = 0; buf[i] != NULL; i++)
129         if (buf[i] == ':') {
130             delim[colons] = i;
131             ++colons;
132         }
134     if (colons != 6) {
135         error(ERROR1);
136         continue;
137     }
138     delim[6] = i - 1;
139     delim[7] = NULL;
141     /*
142     * Check the first char is alpha; the rest alphanumeric;
143     * and that the name does not consist solely of uppercase
144     * alpha chars
145     */
146     if (buf[0] == ':')
147         error(ERROR2b);
148     else if (!isalpha(buf[0]))
149         error(ERROR2a);
151     for (i = 0; buf[i] != ':'; i++) {
152         if (!isalnum(buf[i]) &&
153             buf[i] != '-' &&
154             buf[i] != '_' &&
155             buf[i] != '.')
156             badc++;
157         else if (islower(buf[i]))
158             lc++;
159     }
160     if (lc == 0)
161         error(ERROR2c);
162     if (badc > 0)
163         error(ERROR2);
165     /* Check for valid number of characters in logname */
167     if (i <= 0 || i > LOGNAME_MAX)
168         error(ERROR3);
170     /* Check that UID is numeric and <= MAXUID */
172     errno = 0;
173     str = &buf[delim[1] + 1];
174     uid = strtol(str, &lastc, 10);
175     if (lastc != str + (delim[2] - delim[1]) - 1 ||
176         uid > MAXUID || errno == ERANGE)
177         error(ERROR4);
179     /* Check that GID is numeric and <= MAXUID */
181     errno = 0;
182     str = &buf[delim[2] + 1];
183     gid = strtol(str, &lastc, 10);
184     if (lastc != str + (delim[3] - delim[2]) - 1 ||
185         gid > MAXUID || errno == ERANGE)
186         error(ERROR5);
188     /* Check initial working directory */

```

```

190         for (j = 0, i = (delim[4] + 1); i < delim[5]; j++, i++)
191             logbuf[j] = buf[i];
192         logbuf[j] = '\0';
194         if (logbuf[0] == NULL)
195             error(ERROR6a);
196         else if ((stat(logbuf, &obuf)) == -1)
197             error(ERROR6);
199     /* Check program to use as shell */
201     if ((buf[(delim[5] + 1)]) != '\n') {
203         for (j = 0, i = (delim[5] + 1); i < delim[6]; j++, i++)
204             logbuf[j] = buf[i];
205         logbuf[j] = '\0';
207         if (strcmp(logbuf, "**") == 0) /* subsystem login */
208             continue;
210         if ((stat(logbuf, &obuf)) == -1)
211             error(ERROR7);
213         for (j = 0; j < 512; j++)
214             logbuf[j] = NULL;
215     }
216     }
217     (void) fclose(fp);
218     return (code);
219 }

```

unchanged portion omitted

```

*****
4517 Sat Jun 1 10:02:49 2013
new/usr/src/cmd/sgs/Makefile
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 # Copyright (c) 2013 Gary Mills
22 #
23 # Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
24 #

26 include $(SRC)/cmd/Makefile.cmd

28 # Note: Why SUBDIRS-common isn't sorted alphabetically
29 #
30 # The items under SGS are not independent of each other.
31 # They must be built in an order that ensures that
32 # all dependencies of an item have been built before the
33 # item itself.
34 #
35 SUBDIRS-common= libconv \
36 .WAIT \
37 libdl \
38 libelf \
39 liblddbg \
40 .WAIT \
41 libld \
42 libldmake \
43 libldstab \
44 librtld \
45 libcrle \
46 .WAIT \
47 0@0 \
48 getloginx \
49 ld \
50 ldd \
51 lddstab \
52 rtld \
53 link_audit \
54 .WAIT \
55 librtld_db \
56 ldprof \
57 pvs \
58 crle \
59 ar \
60 dump \

```

```

61 elfdump \
62 elfedit \
63 elfwrap \
64 error \
65 gprof \
66 lari \
67 lex \
68 lorder \
69 m4 \
70 mcs \
71 moe \
72 nm \
73 prof \
74 ranlib \
75 size \
76 symorder \
77 tsort \
78 undef \
79 yacc

81 SUBDIRS-i386=
82 SUBDIRS-sparc= rtld.4.x

84 SUBDIRS= $(SUBDIRS-common) $(SUBDIRS-$(MACH))

86 # Messaging support
87 #
88 POSUBDIRS= m4 nm tsort yacc
89 POFILE= sgs.po
90 POFILES= $(POSUBDIRS:%=%/%.po)

92 MSGSUBDIRS= ld ldd libld liblddbg \
93 libldstab librtld libelf \
94 ldprof libcrle pvs elfdump \
95 elfedit crle moe lari \
96 librtld_db elfwrap ar

98 MSGDIR= messages

101 all := TARGET= all
102 install := TARGET= install
103 clean := TARGET= clean
104 clobber := TARGET= clobber
105 delete := TARGET= delete
106 lint := TARGET= lint
107 _msg := TARGET= catalog
108 _msg_gettext := TARGET= catalog
109 _msg_sgsmg := TARGET= catalog
110 chkmsg := TARGET= chkmsg

113 .KEEP_STATE:

115 .PARALLEL: $(SUBDIRS)

117 all install: native-add .WAIT $(SUBDIRS)

119 include $(SRC)/cmd/Makefile.targ

121 # Messaging support
122 #
123 _msg: _msg_gettext _msg_sgsmg

125 _msg_gettext: $(MSGDOMAIN)/$(POFILE)

```

## new/usr/src/cmd/sgs/Makefile

3

```

127 _msg_sgsmmsg: $(MSGDIR)
129 $(MSGDOMAIN)/$(POFILE): \
130     $(MSGDOMAIN) $(POFILE)
132 $(POFILE):      $(POSUBDIRS)
133                 $(RM) $(POFILE)
134                 cat $(POFILES) > $(POFILE)
136 $(MSGDIR):      $(MSGSUBDIRS) FRC
137                 @ cd $@; pwd; $(MAKE) $(TARGET)
139 chkmsg:         libconv $(MSGSUBDIRS) FRC
141 check:          chkmsg
143 # built from lib/Makefile
144 install_lib:    FRC
145                 @ cd lex; pwd; $(MAKE) $@
146                 @ cd yacc; pwd; $(MAKE) $@
148 lint:           $(SUBDIRS)
150 delete \
151 clean clobber: native-clobber .WAIT $(SUBDIRS) $(MSGDIR)
153 $(SUBDIRS):     FRC
154                 @ cd $@; pwd; $(MAKE) $(TARGET)
157 # Integration of ld and ld.so.1 in some development cycles requires that both
158 # of these modules be built using the new ld.  This 'native' target allows us
159 # to build a local ld which will then be used to build the delivered version of
160 # itself and ld.so.1.  Once this new functionality appears in the standard ld
161 # this target can be disabled.
163 native-add:     native-proto FRC
164                 @ cd tools/$(MACH); pwd; $(MAKE) native
165                 @ cd libconv/$(MACH); pwd; $(MAKE)
166                 @ cd libelf/$(MACH); pwd; $(MAKE) native
167                 @ cd liblddbg/$(MACH); pwd; $(MAKE) native
168                 @ cd libldstab/$(MACH); pwd; $(MAKE) native
169                 @ cd libld/$(MACH); pwd; $(MAKE) native
170                 @ cd ld/$(MACH); pwd; $(MAKE) native
172 native-clobber:
173                 @ cd tools; pwd; $(MAKE) $(TARGET)
174                 $(RM) -r proto/$(MACH)
176 native-proto:
177                 -@mkdir -p proto/$(MACH)
179 FRC:
181 #
182 # Cross-reference customization: ignore the directories named by XRPRUNE,
183 # and tweak the file globs slightly.
184 #
185 XRPRUNE=        rtld.4.x packages abi
186 XRADD=          *.msg mapfile* llib-[a-z]*
187 XRDEL=          Makefile* kobj_*
189 #
190 # Establish a set of directories for xref to search.  As there are duplicates
191 # of things like headers, and only one file will be added to the xref database,
192 # we want xref to list the source file.

```

## new/usr/src/cmd/sgs/Makefile

4

```

193 #
194 XDIRS=          . \
195                 ../../common/elfcap \
196                 ../../head \
197                 ../../uts/common/krtld \
198                 ../../uts/common/sys \
199                 ../../uts/sparc/sys \
200                 ../../uts/sparc/krtld \
201                 ../../uts/intel/ia32/krtld \
202                 ../../uts/intel/amd64/krtld
204 xref:           FRC
205                 @ $(RM) cscope.*
206                 xref -p -x cscope

```

new/usr/src/cmd/sgs/getloginx/Makefile

1

\*\*\*\*\*

964 Sat Jun 1 10:02:49 2013

new/usr/src/cmd/sgs/getloginx/Makefile

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 #
23 # Copyright (c) 2013 Gary Mills
24 #
25 # Copyright (c) 1996 by Sun Microsystems, Inc.
26 # All rights reserved.
28 include $(SRC)/cmd/sgs/Makefile.sub
```

new/usr/src/cmd/sgs/getloginx/Makefile.com

1

\*\*\*\*\*

1261 Sat Jun 1 10:02:49 2013

new/usr/src/cmd/sgs/getloginx/Makefile.com

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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 # Copyright (c) 2013 Gary Mills
22 #
23 # Copyright 2006 Sun Microsystems, Inc. All rights reserved.
24 # Use is subject to license terms.
25 #
26 #
28 LIBRARY=      getloginx.a
29 VERS=        .1
31 OBJECTS=     getloginx.o
33 include      $(SRC)/lib/Makefile.lib
35 MAPFILES=    ../common/mapfile-vers
36 LDLIBS +=    -lc
37 BUILD.SO=    $(LD) -o $@ -G $(DYNFLAGS) $(PICS) $(LDLIBS)
39 SRCS=        $(OBJECTS:%.o=../common/%.c)
41 CLEANFILES += $(LINTOUT)
42 CLOBBERFILES += $(DYNLIB)
44 ROOTDYNLIB=  $(DYNLIB:%=$(ROOTLIBDIR)/%)
```

new/usr/src/cmd/sgs/getloginx/Makefile.targ

1

\*\*\*\*\*

1123 Sat Jun 1 10:02:49 2013

new/usr/src/cmd/sgs/getloginx/Makefile.targ

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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 #
24 # Copyright 2008 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #

28 pics/%.o:      ../common/%.c
29                $(COMPILE.c) -o $@ $<
30                $(POST_PROCESS_O)

32 all:           $(DYNLIB)

34 $(DYNLIB):     pics .WAIT

36 include        $(SRC)/lib/Makefile.targ

38 delete:
39                $(RM) $(DYNLIB)

41 lint:          lintcheck
```



new/usr/src/cmd/sgs/getloginx/amd64/Makefile

1

\*\*\*\*\*

1177 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/amd64/Makefile

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 # Copyright (c) 2013 Gary Mills
23 #
24 # Copyright 2004 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #
27 #

29 include      $(SRC)/cmd/sgs/getloginx/Makefile.com

31 ASFLAGS +=   -D__amd64 $(amd64_ASFLAGS)

33 .KEEP_STATE:

35 install:    all $(ROOTLIBDIR64)/$(DYNLIB)

37 include     $(SRC)/cmd/sgs/getloginx/Makefile.targ
38 include     ../../Makefile.sub.64
```

new/usr/src/cmd/sgs/getloginx/common/getloginx.c

1

\*\*\*\*\*

1195 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/common/getloginx.c

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6  * (the "License"). You may not use this file except in compliance
7  * with the License.
8  *
9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 * or http://www.opensolaris.org/os/licensing.
11 * See the License for the specific language governing permissions
12 * and limitations under the License.
13 *
14 * When distributing Covered Code, include this CDDL HEADER in each
15 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 * If applicable, add the following below this CDDL HEADER, with the
17 * fields enclosed by brackets "[]" replaced with your own identifying
18 * information: Portions Copyright [yyyy] [name of copyright owner]
19 *
20 * CDDL HEADER END
21 */
22 /*
23  * Copyright (c) 2013 Gary Mills
24  *
25  * Copyright (c) 1996 by Sun Microsystems, Inc.
26  *
27  * Truncate a long login name returned by getlogin()
28  * for programs that copy the name to a nine-byte buffer.
29  *
30  * Use:
31  * LD_PRELOAD=getloginx.so.1 program args ...
32  *
33  */

35 extern char *getloginx(void);

37 char *
38 getlogin(void)
39 {
40     return (getloginx());
41 }

43 /* */
```

new/usr/src/cmd/sgs/getloginx/common/mapfile-vers

1

\*\*\*\*\*

1310 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/common/mapfile-vers

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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 #
24 #
25 #
26 # MAPFILE HEADER START
27 #
28 # WARNING: STOP NOW. DO NOT MODIFY THIS FILE.
29 # Object versioning must comply with the rules detailed in
30 #
31 #     usr/src/lib/README.mapfiles
32 #
33 # You should not be making modifications here until you've read the most current
34 # copy of that file. If you need help, contact a gatekeeper for guidance.
35 #
36 # MAPFILE HEADER END
37 #
38 #
39 $mapfile_version 2
40 #
41 SYMBOL_VERSION ILLUMOS_0.1 { # first release of getloginx.so.1
42     global:
43         getlogin;
44     local:
45         *;
46 };
```

new/usr/src/cmd/sgs/getloginx/i386/Makefile

1

\*\*\*\*\*

1093 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/i386/Makefile

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 # Copyright (c) 2013 Gary Mills
23 #
24 # Copyright 2004 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #

28 include      $(SRC)/cmd/sgs/getloginx/Makefile.com

30 .KEEP_STATE:

32 include      $(SRC)/cmd/sgs/getloginx/Makefile.targ

34 install:     all $(ROOTDYNLIB)
```

new/usr/src/cmd/sgs/getloginx/sparc/Makefile

1

\*\*\*\*\*

1093 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/sparc/Makefile

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 # Copyright (c) 2013 Gary Mills
23 #
24 # Copyright 2004 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #

28 include      $(SRC)/cmd/sgs/getloginx/Makefile.com

30 .KEEP_STATE:

32 include      $(SRC)/cmd/sgs/getloginx/Makefile.targ

34 install:     all $(ROOTDYNLIB)
```

new/usr/src/cmd/sgs/getloginx/sparcv9/Makefile

1

\*\*\*\*\*

1136 Sat Jun 1 10:02:50 2013

new/usr/src/cmd/sgs/getloginx/sparcv9/Makefile

2989 Eliminate use of LOGNAME\_MAX in ON

1166 useradd have warning with name more 8 chars

\*\*\*\*\*

```
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, Version 1.0 only
6 # (the "License"). You may not use this file except in compliance
7 # with the License.
8 #
9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 # or http://www.opensolaris.org/os/licensing.
11 # See the License for the specific language governing permissions
12 # and limitations under the License.
13 #
14 # When distributing Covered Code, include this CDDL HEADER in each
15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 # If applicable, add the following below this CDDL HEADER, with the
17 # fields enclosed by brackets "[]" replaced with your own identifying
18 # information: Portions Copyright [yyyy] [name of copyright owner]
19 #
20 # CDDL HEADER END
21 #
22 # Copyright (c) 2013 Gary Mills
23 #
24 # Copyright 2004 Sun Microsystems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #

28 include      $(SRC)/cmd/sgs/getloginx/Makefile.com

30 .KEEP_STATE:

32 install:     all $(ROOTLIBDIR64)/$(DYNLIB)

34 include      $(SRC)/cmd/sgs/getloginx/Makefile.targ
35 include      ../../Makefile.sub.64
```

```

*****
18980 Sat Jun 1 10:02:50 2013
new/usr/src/cmd/w/w.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  */

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

31 /*
32  * University Copyright- Copyright (c) 1982, 1986, 1988
33  * The Regents of the University of California
34  * All Rights Reserved
35  *
36  * University Acknowledgment- Portions of this document are derived from
37  * software developed by the University of California, Berkeley, and its
38  * contributors.
39  */

41 /*
42  * This is the new w command which takes advantage of
43  * the /proc interface to gain access to the information
44  * of all the processes currently on the system.
45  *
46  * This program also implements 'uptime'.
47  *
48  * Maintenance note:
49  *
50  * Much of this code is replicated in whodo.c.  If you're
51  * fixing bugs here, then you should probably fix 'em there too.
52  */

54 #include <stdio.h>
55 #include <string.h>
56 #include <stdarg.h>
57 #include <stdlib.h>
58 #include <ctype.h>
59 #include <fcntl.h>
60 #include <time.h>

```

```

61 #include <errno.h>
62 #include <sys/types.h>
63 #include <utmpx.h>
64 #include <sys/stat.h>
65 #include <dirent.h>
66 #include <procfs.h>          /* /proc header file */
67 #include <locale.h>
68 #include <unistd.h>
69 #include <sys/loadavg.h>
70 #include <limits.h>
71 #include <priv_utils.h>

73 /*
74  * Use the full lengths from utmpx for user and line.
75  * utmpx defines wider fields for user and line.  For compatibility of output,
76  * we are limiting these to the old maximums in utmp.  Define UTMPX_NAMELEN
77  * to use the full lengths.
78  */
79 #ifndef UTMPX_NAMELEN
80 #define UTMPX_NAMELEN 12
81 #endif

82 #define LOGIN_WIDTH 8
83 #define LINE_WIDTH 12

84 #define DIV60(t) ((t+30)/60) /* x/60 rounded */

86 #ifdef ERR
87 #undef ERR
88 #endif
89 #define ERR (-1)

91 #define HSIZE 256 /* size of process hash table */
92 #define PROC_DIR "/proc"
93 #define INITPROC (pid_t)1 /* init process pid */
94 #define NONE 'n' /* no state */
95 #define RUNNING 'r' /* runnable process */
96 #define ZOMBIE 'z' /* zombie process */
97 #define VISITED 'v' /* marked node as visited */
98 #define PRINTF(a) if (printf a < 0) { \
99     perror((gettext("%s: printf failed"), prog)); \
100    exit(1); }

102 struct uproc {
103     pid_t p_upid; /* process id */
104     char p_state; /* numeric value of process state */
105     dev_t p_ttyd; /* controlling tty of process */
106     time_t p_time; /* seconds of user & system time */
107     time_t p_ctime; /* seconds of child user & sys time */
108     int p_igintr; /* 1 = ignores SIGQUIT and SIGINT */
109     char p_comm[PRARGSZ+1]; /* command */
110     char p_args[PRARGSZ+1]; /* command line arguments */
111     struct uproc *p_child; /* first child pointer */
112     struct uproc *p_sibling; /* sibling pointer */
113     pgrp_t p_pgrp; /* pgrp link */
114     struct uproc *p_link; /* hash table chain pointer */
115 };

117 /*

```

```

118 *      define hash table for struct uproc
119 *      Hash function uses process id
120 *      and the size of the hash table(HSIZE)
121 *      to determine process index into the table.
122 */
123 static struct uproc      pr_htbl[HSIZE];

125 static struct      uproc      *findhash(pid_t);
126 static time_t      findidle(char *);
127 static void        clnarglist(char *);
128 static void        showtotals(struct uproc *);
129 static void        calctotals(struct uproc *);
130 static void        prttime(time_t, char *);
131 static void        prtat(time_t *time);
132 static void        checkampm(char *str);

134 static char        *prog;          /* pointer to invocation name */
135 static int         header = 1;     /* true if -h flag: don't print heading */
136 static int         lflag = 1;     /* set if -l flag; 0 for -s flag: short form */
137 static char        *sel_user;     /* login of particular user selected */
138 static char        firstchar;     /* first char of name of prog invoked as */
139 static int         login;         /* true if invoked as login shell */
140 static time_t      now;           /* current time of day */
141 static time_t      uptime;        /* time of last reboot & elapsed time since */
142 static int         nusers;        /* number of users logged in now */
143 static time_t      idle;          /* number of minutes user is idle */
144 static time_t      jobtime;       /* total cpu time visible */
145 static char        doing[520];    /* process attached to terminal */
146 static time_t      proctime;      /* cpu time of process in doing */
147 static pid_t      curpid, empty;
148 static int         add_times;     /* boolean: add the cpu times or not */

150 #if SIGQUIT > SIGINT
151 #define ACTSIZE SIGQUIT
152 #else
153 #define ACTSIZE SIGINT
154 #endif

156 int
157 main(int argc, char *argv[])
158 {
159     struct utmpx      *ut;
160     struct utmpx      *utmpbegin;
161     struct utmpx      *utmpend;
162     struct utmpx      *utp;
163     struct uproc      *up, *parent, *pgrp;
164     struct psinfo      info;
165     struct sigaction  actinfo[ACTSIZE];
166     struct pstatus      statinfo;
167     size_t             size;
168     struct stat        sbuf;
169     DIR                *dirp;
170     struct dirent      *dp;
171     char               pname[64];
172     char               *fname;
173     int                procfd;
174     char               *cp;
175     int                i;
176     int                days, hrs, mins;
177     int                entries;
178     double             loadavg[3];

180     /*
181      * This program needs the proc_owner privilege
182      */
183     (void) __init_suid_priv(PU_CLEARLIMITSET, PRIV_PROC_OWNER,

```

```

184         (char *)NULL);

186         (void) setlocale(LC_ALL, "");
187 #if !defined(TEXT_DOMAIN)
188 #define TEXT_DOMAIN "SYS_TEST"
189 #endif
190         (void) textdomain(TEXT_DOMAIN);

192         login = (argv[0][0] == '-');
193         cp = strrchr(argv[0], '/');
194         firstchar = login ? argv[0][1] : (cp == 0) ? argv[0][0] : cp[1];
195         prog = argv[0];

197         while (argc > 1) {
198             if (argv[1][0] == '-') {
199                 for (i = 1; argv[1][i]; i++) {
200                     switch (argv[1][i]) {

202                         case 'h':
203                             header = 0;
204                             break;

206                         case 'l':
207                             lflag++;
208                             break;
209                         case 's':
210                             lflag = 0;
211                             break;

213                         case 'u':
214                         case 'w':
215                             firstchar = argv[1][i];
216                             break;

218                         default:
219                             (void) fprintf(stderr, gettext(
220                                 "%s: bad flag %s\n"),
221                                 prog, argv[1]);
222                             exit(1);
223                     }
224                 }
225             } else {
226                 if (!isalnum(argv[1][0]) || argc > 2) {
227                     (void) fprintf(stderr, gettext(
228                         "usage: %s [ -hlsuw ] [ user ]\n"), prog);
229                     exit(1);
230                 } else
231                     sel_user = argv[1];
232             }
233             argc--; argv++;
234         }

236         /*
237          * read the UTMP_FILE (contains information about each logged in user)
238          */
239         if (stat(UTMPX_FILE, &sbuf) == ERR) {
240             (void) fprintf(stderr, gettext("%s: stat error of %s: %s\n"),
241                 prog, UTMPX_FILE, strerror(errno));
242             exit(1);
243         }
244         entries = sbuf.st_size / sizeof (struct futmpx);
245         size = sizeof (struct utmpx) * entries;
246         if ((ut = malloc(size)) == NULL) {
247             (void) fprintf(stderr, gettext("%s: malloc error of %s: %s\n"),
248                 prog, UTMPX_FILE, strerror(errno));
249             exit(1);

```



```

250     }
252     (void) utmpxname(UTMPX_FILE);
254     utmpbegin = ut;
255     utmpend = (struct utmpx *)((char *)utmpbegin + size);
257     setutxent();
258     while ((ut < utmpend) && ((utp = getutxent()) != NULL))
259         (void) memcpy(ut++, utp, sizeof (*ut));
260     endutxent();
262     (void) time(&now);      /* get current time */
264     if (header) { /* print a header */
265         prtat(&now);
266         for (ut = utmpbegin; ut < utmpend; ut++) {
267             if (ut->ut_type == USER_PROCESS) {
268                 if (!nonuser(*ut))
269                     nusers++;
270             } else if (ut->ut_type == BOOT_TIME) {
271                 uptime = now - ut->ut_xtime;
272                 uptime += 30;
273                 days = uptime / (60*60*24);
274                 uptime %= (60*60*24);
275                 hrs = uptime / (60*60);
276                 uptime %= (60*60);
277                 mins = uptime / 60;
279                 PRINTF((gettext(" up")));
280                 if (days > 0)
281                     PRINTF((gettext(
282                         " %d day(s)", days));
283                 if (hrs > 0 && mins > 0) {
284                     PRINTF((" %2d:%02d", hrs, mins));
285                 } else {
286                     if (hrs > 0)
287                         PRINTF((gettext(
288                             " %d hr(s)", hrs));
289                     if (mins > 0)
290                         PRINTF((gettext(
291                             " %d min(s)", mins));
292                 }
293             }
294         }
296         ut = utmpbegin; /* rewind utmp data */
297         PRINTF(((nusers == 1) ?
298             gettext(" %d user") : gettext(" %d users")), nusers));
299         /*
300          * Print 1, 5, and 15 minute load averages.
301          */
302         (void) getloadavg(loadavg, 3);
303         PRINTF((gettext(", load average: %.2f, %.2f, %.2f\n"),
304             loadavg[LOADAVG_1MIN], loadavg[LOADAVG_5MIN],
305             loadavg[LOADAVG_15MIN]));
307         if (firstchar == 'u') /* uptime command */
308             exit(0);
310         if (lflag) {
311             PRINTF((dcgettext(NULL, "User      tty
312             "login@ idle JCPU PCPU what\n", LC_TIME)));
313         } else {
314             PRINTF((dcgettext(NULL,
315             "User      tty          idle  what\n", LC_TIME)));

```

```

316     }
318     if (fflush(stdout) == EOF) {
319         perror((gettext("%s: fflush failed\n"), prog));
320         exit(1);
321     }
322 }
324 /*
325  * loop through /proc, reading info about each process
326  * and build the parent/child tree
327  */
328 if (!(dirp = opendir(PROCDIR))) {
329     (void) fprintf(stderr, gettext("%s: could not open %s: %s\n"),
330         prog, PROCDIR, strerror(errno));
331     exit(1);
332 }
334 while ((dp = readdir(dirp)) != NULL) {
335     if (dp->d_name[0] == '.')
336         continue;
337 retry:
338     (void) sprintf(pname, "%s/%s/", PROCDIR, dp->d_name);
339     fname = pname + strlen(pname);
340     (void) strcpy(fname, "psinfo");
341     if ((procfd = open(pname, O_RDONLY)) < 0)
342         continue;
343     if (read(procfd, &info, sizeof (info)) != sizeof (info)) {
344         int err = errno;
345         (void) close(procfd);
346         if (err == EAGAIN)
347             goto retry;
348         if (err != ENOENT)
349             (void) fprintf(stderr, gettext(
350                 "%s: read() failed on %s: %s\n"),
351                 prog, pname, strerror(err));
352         continue;
353     }
354     (void) close(procfd);
356     up = findhash(info.pr_pid);
357     up->p_ttyd = info.pr_ttydev;
358     up->p_state = (info.pr_nlwp == 0? ZOMBIE : RUNNING);
359     up->p_time = 0;
360     up->p_ctime = 0;
361     up->p_igintr = 0;
362     (void) strncpy(up->p_comm, info.pr_fname,
363         sizeof (info.pr_fname));
364     up->p_args[0] = 0;
366     if (up->p_state != NONE && up->p_state != ZOMBIE) {
367         (void) strcpy(fname, "status");
369         /* now we need the proc_owner privilege */
370         (void) __priv_bracket(PRIV_ON);
372         procfd = open(pname, O_RDONLY);
374         /* drop proc_owner privilege after open */
375         (void) __priv_bracket(PRIV_OFF);
377         if (procfd < 0)
378             continue;
380         if (read(procfd, &statinfo, sizeof (statinfo))
381             != sizeof (statinfo)) {

```

```

382         int err = errno;
383         (void) close(procfd);
384         if (err == EAGAIN)
385             goto retry;
386         if (err != ENOENT)
387             (void) fprintf(stderr, gettext(
388                 "%s: read() failed on %s: %s\n"),
389                 prog, pname, strerror(err));
390         continue;
391     }
392     (void) close(procfd);
393
394     up->p_time = statinfo.pr_utime.tv_sec +
395         statinfo.pr_stime.tv_sec; /* seconds */
396     up->p_ctime = statinfo.pr_cutime.tv_sec +
397         statinfo.pr_cstime.tv_sec;
398
399     (void) strcpy(fname, "sigact");
400
401     /* now we need the proc_owner privilege */
402     (void) __priv_bracket(PRIV_ON);
403
404     procfd = open(pname, O_RDONLY);
405
406     /* drop proc_owner privilege after open */
407     (void) __priv_bracket(PRIV_OFF);
408
409     if (procfd < 0)
410         continue;
411
412     if (read(procfd, actinfo, sizeof (actinfo))
413         != sizeof (actinfo)) {
414         int err = errno;
415         (void) close(procfd);
416         if (err == EAGAIN)
417             goto retry;
418         if (err != ENOENT)
419             (void) fprintf(stderr, gettext(
420                 "%s: read() failed on %s: %s\n"),
421                 prog, pname, strerror(err));
422         continue;
423     }
424     (void) close(procfd);
425
426     up->p_igintr =
427         actinfo[SIGINT-1].sa_handler == SIG_IGN &&
428         actinfo[SIGQUIT-1].sa_handler == SIG_IGN;
429
430     /*
431     * Process args.
432     */
433     up->p_args[0] = 0;
434     clnarglist(info.pr_psargs);
435     (void) strcat(up->p_args, info.pr_psargs);
436     if (up->p_args[0] == 0 ||
437         up->p_args[0] == '-' && up->p_args[1] <= ' ' ||
438         up->p_args[0] == '?') {
439         (void) strcat(up->p_args, " (");
440         (void) strcat(up->p_args, up->p_comm);
441         (void) strcat(up->p_args, ")");
442     }
443 }
444
445 /*
446 * link pgrp together in case parents go away
447 * Pgrp chain is a single linked list originating

```

```

448     * from the pgrp leader to its group member.
449     */
450     if (info.pr_pgid != info.pr_pid) { /* not pgrp leader */
451         pgrp = findhash(info.pr_pgid);
452         up->p_pgrppl = pgrp->p_pgrppl;
453         pgrp->p_pgrppl = up;
454     }
455     parent = findhash(info.pr_ppid);
456
457     /* if this is the new member, link it in */
458     if (parent->p_upid != INITPROCESS) {
459         if (parent->p_child) {
460             up->p_sibling = parent->p_child;
461             up->p_child = 0;
462         }
463         parent->p_child = up;
464     }
465 }
466
467 /* revert to non-privileged user after opening */
468 (void) __priv_relinquish();
469
470 (void) closedir(dirp);
471 (void) time(&now); /* get current time */
472
473 /*
474 * loop through utmpx file, printing process info
475 * about each logged in user
476 */
477 for (ut = utmpbegin; ut < utmpend; ut++) {
478     if (ut->ut_type != USER_PROCESS)
479         continue;
480     if (sel_user && strcmp(ut->ut_name, sel_user, NMAX) != 0)
481         continue; /* we're looking for somebody else */
482
483     /* print login name of the user */
484     PRINTF(("%-*.s", LOGIN_WIDTH, NMAX, ut->ut_name));
485     PRINTF(("%-*.s", NMAX, NMAX, ut->ut_name));
486
487     /* print tty user is on */
488     if (lflag) {
489         PRINTF(("%-*.s", LINE_WIDTH, LMAX, ut->ut_line));
490         PRINTF(("%-*.s", LMAX, LMAX, ut->ut_line));
491     } else {
492         if (ut->ut_line[0] == 'p' && ut->ut_line[1] == 't' &&
493             ut->ut_line[2] == 's' && ut->ut_line[3] == '/') {
494             PRINTF(("%-*.3s", LMAX, &ut->ut_line[4]));
495         } else {
496             PRINTF(("%-*.s", LINE_WIDTH, LMAX,
497                 ut->ut_line));
498             PRINTF(("%-*.s", LMAX, LMAX, ut->ut_line));
499         }
500     }
501
502     /* print when the user logged in */
503     if (lflag) {
504         time_t tim = ut->ut_xtime;
505         prtat(&tim);
506     }
507
508     /* print idle time */
509     idle = findidle(ut->ut_line);
510     if (idle >= 36 * 60) {
511         PRINTF((dcgettext(NULL, "%2ddays", LC_TIME),
512             (idle + 12 * 60) / (24 * 60)));
513     } else

```

```
511         prttime(idle, " ");
512         showtotals(findhash(ut->ut_pid));
513     }
514     if (fclose(stdout) == EOF) {
515         perror((gettext("%s: fclose failed"), prog));
516         exit(1);
517     }
518     return (0);
519 }
_____unchanged_portion_omitted_____
```

new/usr/src/cmd/wall/wall.c

1

```
*****
11107 Sat Jun 1 10:02:50 2013
new/usr/src/cmd/wall/wall.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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, Version 1.0 only
6  * (the "License").  You may not use this file except in compliance
7  * with the License.
8  *
9  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
10 * or http://www.opensolaris.org/os/licensing.
11 * See the License for the specific language governing permissions
12 * and limitations under the License.
13 *
14 * When distributing Covered Code, include this CDDL HEADER in each
15 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
16 * If applicable, add the following below this CDDL HEADER, with the
17 * fields enclosed by brackets "[]" replaced with your own identifying
18 * information: Portions Copyright [yyyy] [name of copyright owner]
19 *
20 * CDDL HEADER END
21 */
22 /*      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T */
23 /*      All Rights Reserved      */

27 /*
28 * Copyright 1988-2003 Sun Microsystems, Inc.  All rights reserved.
29 * Use is subject to license terms.
30 */

32 /*
33 * Copyright 2012 Joyent, Inc. All rights reserved.
34 *
35 * Copyright (c) 2013 Gary Mills
36 */

38 #include <signal.h>
39 #include <stdio.h>
40 #include <stdlib.h>
41 #include <grp.h>
42 #include <sys/types.h>
43 #include <unistd.h>
44 #include <string.h>
45 #include <ctype.h>
46 #include <sys/stat.h>
47 #include <utmpx.h>
48 #include <sys/utsname.h>
49 #include <dirent.h>
50 #include <pwd.h>
51 #include <fcntl.h>
52 #include <time.h>
53 #include <errno.h>
54 #include <locale.h>
55 #include <syslog.h>
56 #include <sys/wait.h>
57 #include <limits.h>
58 #include <libzonecfg.h>
59 #include <zone.h>
60 #include <sys/contract/process.h>
```

new/usr/src/cmd/wall/wall.c

2

```
61 #include <libcontract.h>
62 #include <sys/ctfs.h>

64 /*
65 * Use the full lengths from utmpx for user and line.
66 * utmpx defines wider fields for user and line.  For compatibility of output,
67 * we are limiting these to the old maximums in utmp.  Define UTMPX_NAMELEN
68 * to use the full lengths.
69 */
70 #define NMAX      (sizeof (((struct utmpx *)0)->ut_user))
71 #define LMAX      (sizeof (((struct utmpx *)0)->ut_line))
72 #ifndef UTMPX_NAMELEN
73 #define XXX - utmp -fix name length */
74 #define NMAX      (_POSIX_LOGIN_NAME_MAX - 1)
75 #define LMAX      12
76 #else /* UTMPX_NAMELEN */
77 #define NMAX      (sizeof (((struct utmpx *)0)->ut_user))
78 #define LMAX      (sizeof (((struct utmpx *)0)->ut_line))
79 #endif /* UTMPX_NAMELEN */

80 static char      mesg[3000];
81 static char      *infile;
82 static int       gflag;
83 static struct    group *pgrp;
84 static char      *grpname;
85 static char      line[MAXNAMLEN+1] = "???";
86 static char      syst[MAXNAMLEN+1];
87 static time_t    tloc;
88 static struct    utsname utsn;
89 static char      who[NMAX+1] = "???";
90 static char      who[9] = "???";
91 static char      time_buf[50];
92 #define DATE_FMT      "%a %b %e %H:%M:%S"

93 static void sendmes(struct utmpx *, zoneid_t);
94 static void sendmes_tozone(zoneid_t, int);
95 static int chkgrp(char *);
96 static char *copy_str_till(char *, char *, char, int);

97 static int init_template(void);
98 int contract_abandon_id(ctid_t);

99 int
100 main(int argc, char *argv[])
101 {
102     FILE *f;
103     char *ptr, *start;
104     struct passwd *pwd;
105     char *term_name;
106     int c;
107     int aflag = 0;
108     int errflg = 0;
109     int zflg = 0;
110     int Zflg = 0;

111     char *zonename = NULL;
112     zoneid_t *zoneidlist = NULL;
113     uint_t nzids_saved, nzids = 0;

114     (void) setlocale(LC_ALL, "");

115     while ((c = getopt(argc, argv, "g:az:Z")) != EOF)
116         switch (c) {
117             case 'a':
118                 aflag++;
119                 break;
120         }
```

```

115     case 'g':
116         if (gflag) {
117             (void) fprintf(stderr,
118                 "Only one group allowed\n");
119             return (1);
120         }
121         if ((pgrp = getgrnam(grpname = optarg)) == NULL) {
122             (void) fprintf(stderr, "Unknown group %s\n",
123                 grpname);
124             return (1);
125         }
126         gflag++;
127         break;
128     case 'z':
129         zflag++;
130         zonename = optarg;
131         if (getzoneidbyname(zonename) == -1) {
132             (void) fprintf(stderr, "Specified zone %s
133                 "is invalid", zonename);
134             return (1);
135         }
136         break;
137     case 'Z':
138         Zflag++;
139         break;
140     case '?':
141         errflg++;
142         break;
143     }
144
145     if (errflg) {
146         (void) fprintf(stderr,
147             "Usage: wall [-a] [-g group] [-z zone] [-Z] [files...]\n");
148         return (1);
149     }
150
151     if (zflag && Zflag) {
152         (void) fprintf(stderr, "Cannot use -z with -Z\n");
153         return (1);
154     }
155
156     if (optind < argc)
157         infile = argv[optind];
158
159     if (uname(&utsn) == -1) {
160         (void) fprintf(stderr, "wall: uname() failed, %s\n",
161             strerror(errno));
162         return (2);
163     }
164     (void) strcpy(system, utsn.nodename);
165
166     /*
167     * Get the name of the terminal wall is running from.
168     */
169
170     if ((term_name = ttyname(fileno(stderr))) != NULL) {
171         /*
172         * skip the leading "/dev/" in term_name
173         */
174         (void) strncpy(line, &term_name[5], sizeof (line) - 1);
175     }
176
177     if (who[0] == '?') {
178         if (pwd = getpwuid(getuid()))
179             (void) strncpy(&who[0], pwd->pw_name, sizeof (who));
180     }

```

```

182     f = stdin;
183     if (infile) {
184         f = fopen(infile, "r");
185         if (f == NULL) {
186             (void) fprintf(stderr, "Cannot open %s\n", infile);
187             return (1);
188         }
189     }
190
191     start = &mesg[0];
192     ptr = start;
193     while ((ptr - start) < 3000) {
194         size_t n;
195
196         if (fgets(ptr, &mesg[sizeof (mesg)] - ptr, f) == NULL)
197             break;
198         if ((n = strlen(ptr)) == 0)
199             break;
200         ptr += n;
201     }
202     (void) fclose(f);
203
204     /*
205     * If the request is from the rwall daemon then use the caller's
206     * name and host. We determine this if all of the following is true:
207     * 1) First 5 characters are "From "
208     * 2) Next non-white characters are of the form "name@host:"
209     */
210     if (strncmp(line, "???" ) == 0) {
211         char rwho[MAXNAMLEN+1];
212         char rsystem[MAXNAMLEN+1];
213         char *cp;
214
215         if (strncmp(mesg, "From ", 5) == 0) {
216             cp = &mesg[5];
217             cp = copy_str_till(rwho, cp, '@', MAXNAMLEN + 1);
218             if (rwho[0] != '\0') {
219                 cp = copy_str_till(rsystem, ++cp, ':',
220                     MAXNAMLEN + 1);
221                 if (rsystem[0] != '\0') {
222                     (void) strcpy(system, rsystem);
223                     (void) strncpy(rwho, who,
224                         sizeof (who));
225                     (void) strncpy(rwho, who, 9);
226                     (void) strcpy(line, "rpc.rwalld");
227                 }
228             }
229         }
230         (void) time(&tloc);
231         (void) strftime(time_buf, sizeof (time_buf),
232             DATE_FMT, localtime(&tloc));
233
234         if (zflag != 0) {
235             if ((zoneidlist =
236                 malloc(sizeof (zoneid_t))) == NULL ||
237                 (*zoneidlist = getzoneidbyname(zonename)) == -1)
238                 return (errno);
239             nzids = 1;
240         } else if (Zflag != 0) {
241             if (zone_list(NULL, &nzids) != 0)
242                 return (errno);
243         }
244         nzids *= 2;
245         if ((zoneidlist = malloc(nzids * sizeof (zoneid_t))) == NULL)

```

```

246         exit(errno);
247     nzids_saved = nzids;
248     if (zone_list(zoneidlist, &nzids) != 0) {
249         (void) free(zoneidlist);
250         return (errno);
251     }
252     if (nzids > nzids_saved) {
253         free(zoneidlist);
254         goto again;
255     }
256 }
257 if (zflg || Zflg) {
258     for (; nzids > 0; --nzids)
259         sendmes_tozone(zoneidlist[nzids-1], aflag);
260     free(zoneidlist);
261 } else
262     sendmes_tozone(getzoneid(), aflag);
264     return (0);
265 }

```

unchanged portion omitted

```

329 /*
330 * Note to future maintainers: with the change of wall to use the
331 * getutxent() API, the forked children (created by this function)
332 * must call _exit as opposed to exit. This is necessary to avoid
333 * unwanted fflushing of getutxent's stdio stream (caused by atexit
334 * processing).
335 */
336 static void
337 sendmes(struct utmpx *p, zoneid_t zid)
338 {
339     int i;
340     char *s;
341     static char device[LMAX + 6];
342     char *bp;
343     int ibp;
344     FILE *f;
345     int fd, tmpl_fd;
346     boolean_t zoneenter = B_FALSE;
348     if (zid != getzoneid()) {
349         zoneenter = B_TRUE;
350         tmpl_fd = init_template();
351         if (tmpl_fd == -1) {
352             (void) fprintf(stderr, "Could not initialize "
353                 "process contract");
354             return;
355         }
356     }
358     while ((i = (int)fork()) == -1) {
359         (void) alarm(60);
360         (void) wait((int *)0);
361         (void) alarm(0);
362     }
364     if (i)
365         return;
367     if (zoneenter && zone_enter(zid) == -1) {
368         char zonename[ZONENAME_MAX];
369         (void) getzonenamebyid(zid, zonename, ZONENAME_MAX);
370         (void) fprintf(stderr, "Could not enter zone "
371             "%s\n", zonename);
372     }

```

```

373     if (zoneenter)
374         (void) ct_tmpl_clear(tmpl_fd);
376     if (gflag)
377         if (!chkgrp(p->ut_user))
378             _exit(0);
380     (void) signal(SIGHUP, SIG_IGN);
381     (void) alarm(60);
382     s = &device[0];
383     (void) snprintf(s, sizeof (device), "/dev/%.*s", LMAX, p->ut_line);
385     /* check if the device is really a tty */
386     if ((fd = open(s, O_WRONLY|O_NOCTTY|O_NONBLOCK)) == -1) {
387         (void) fprintf(stderr, "Cannot send to %.*s on %s\n",
388             NMAX, p->ut_user, s);
389         perror("open");
390         (void) fflush(stderr);
391         _exit(1);
392     } else {
393         if (!isatty(fd)) {
394             (void) fprintf(stderr,
395                 "Cannot send to device %.*s %s\n",
396                 LMAX, p->ut_line,
397                 "because it's not a tty");
398             openlog("wall", 0, LOG_AUTH);
399             syslog(LOG_CRIT, "%.*s in utmpx is not a tty\n",
400                 LMAX, p->ut_line);
401             closelog();
402             (void) fflush(stderr);
403             _exit(1);
404         }
405     }
406 #ifdef DEBUG
407     (void) close(fd);
408     f = fopen("wall.debug", "a");
409 #else
410     f = fdopen(fd, "w");
411 #endif
412     if (f == NULL) {
413         (void) fprintf(stderr, "Cannot send to %.*s on %s\n",
414             NMAX, &p->ut_user[0], s);
415         perror("open");
416         (void) fflush(stderr);
417         _exit(1);
418     }
419     (void) fprintf(f,
420         "\07\07\07Broadcast Message from %s (%s) on %s %19.19s",
421         who, line, system, time_buf);
422     if (gflag)
423         (void) fprintf(f, " to group %s", grpname);
424     (void) fprintf(f, "... \n");
425 #ifdef DEBUG
426     (void) fprintf(f, "DEBUG: To %.*s on %s\n", NMAX, p->ut_user, s);
427     (void) fprintf(f, "DEBUG: To %.*s on %s\n", p->ut_user, s);
428 #endif
429     i = strlen(msg);
430     for (bp = msg; --i >= 0; bp++) {
431         ibp = (unsigned int)((unsigned char) *bp);
432         if (*bp == '\n')
433             (void) putc('\r', f);
434         if (isprint(ibp) || *bp == '\r' || *bp == '\013' ||
435             *bp == ' ' || *bp == '\t' || *bp == '\n' || *bp == '\007') {
436             (void) putc(*bp, f);
437         } else {
438             if (!isascii(*bp)) {

```

```
438         (void) fputs("M-", f);
439         *bp = toascii(*bp);
440     }
441     if (iscntrl(*bp)) {
442         (void) putc('^', f);
443         (void) putc(*bp + 0100, f);
444     }
445     else
446         (void) putc(*bp, f);
447 }
449 if (*bp == '\n')
450     (void) fflush(f);
452 if (ferror(f) || feof(f)) {
453     (void) printf("\n\007Write failed\n");
454     (void) fflush(stdout);
455     _exit(1);
456 }
457 }
458 (void) fclose(f);
459 (void) close(fd);
460 _exit(0);
461 }
```

```
464 static int
465 chkgrp(char *name)
466 {
467     int i;
468     char user[NMAX + 1];
469     char *p;
470     (void) strncpy(user, name, NMAX);
471     user[NMAX] = '\0';
472     for (i = 0; pgrp->gr_mem[i] && pgrp->gr_mem[i][0]; i++) {
473         if (strcmp(user, pgrp->gr_mem[i]) == 0)
474             for (p = name; *p && *p != ' '; p++)
475                 *p = 0;
476         if (strncmp(name, pgrp->gr_mem[i], 8) == 0)
477             return (1);
478     }
479     return (0);
480 }
```

unchanged portion omitted

new/usr/src/cmd/who/who.c

1

```
*****
20484 Sat Jun 1 10:02:50 2013
new/usr/src/cmd/who/who.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 /*      Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T */
22 /*      All Rights Reserved */

25 /*
26  * Copyright (c) 2013 Gary Mills
27  *
28  * Copyright 2006 Sun Microsystems, Inc. All rights reserved.
29  * Use is subject to license terms.
30  */

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

32 /*
33  *      This program analyzes information found in /var/adm/utmpx
34  *
35  *      Additionally information is gathered from /etc/inittab
36  *      if requested.
37  *
38  *
39  *      Syntax:
40  *
41  *      who am i          Displays info on yourself
42  *
43  *      who -a           Displays information about All
44  *                      entries in /var/adm/utmpx
45  *
46  *      who -b           Displays info on last boot
47  *
48  *      who -d           Displays info on DEAD PROCESSES
49  *
50  *      who -H           Displays HEADERS for output
51  *
52  *      who -l           Displays info on LOGIN entries
53  *
54  *      who -m           Same as who am i
55  *
56  *      who -p           Displays info on PROCESSES spawned by init
57  *
58  *      who -q           Displays short information on
```

new/usr/src/cmd/who/who.c

2

```
59  *
60  *
61  *      who -r           Displays info of current run-level
62  *
63  *      who -s           Displays requested info in SHORT form
64  *
65  *      who -t           Displays info on TIME changes
66  *
67  *      who -T           Displays writeability of each user
68  *                      (+ writeable, - non-writeable, ? hung)
69  *
70  *      who -u           Displays LONG info on users
71  *                      who have LOGGED ON
72  */

74 #define      DATE_FMT      "%b %e %H:%M"

76 /*
77  * %b Abbreviated month name
78  * %e Day of month
79  * %H hour (24-hour clock)
80  * %M minute
81  */
82 #include      <errno.h>
83 #include      <fcntl.h>
84 #include      <stdio.h>
85 #include      <string.h>
86 #include      <sys/types.h>
87 #include      <unistd.h>
88 #include      <stdlib.h>
89 #include      <sys/stat.h>
90 #include      <time.h>
91 #include      <utmpx.h>
92 #include      <locale.h>
93 #include      <pwd.h>
94 #include      <limits.h>

96 static void process(void);
97 static void ck_file(char *);
98 static void dump(void);

100 static struct utmpx *utmpp; /* pointer for getutxent() */

102 /*
103  * Use the full lengths from utmpx for user and line.
104  * utmpx defines wider fields for user and line. For compatibility of output,
105  * we are limiting these to the old maximums in utmp. Define UTMPX_NAMELEN
106  * to use the full lengths.
107  */
107 #ifndef UTMPX_NAMELEN
108 /* XXX - utmp - fix name length */
109 #define NMAX      (_POSIX_LOGIN_NAME_MAX - 1)
110 #define LMAX      12
111 #else /* UTMPX_NAMELEN */
112 #define NMAX      (sizeof (utmpp->ut_user))
113 #define LMAX      (sizeof (utmpp->ut_line))
114 #endif

108 /* Print minimum field widths. */
109 #define LOGIN_WIDTH      8
110 #define LINE_WIDTH      12

112 static char      comment[BUFSIZ]; /* holds inittab comment */
113 static char      errmsg[BUFSIZ]; /* used in snprintf for errors */
114 static int      fd; /* file descriptor for inittab */
115 static int      Hopt = 0; /* 1 = who -H */
```



```

116 static char *inittab; /* ptr to inittab contents */
117 static char *iinit; /* index into inittab */
118 static int justme = 0; /* 1 = who am i */
119 static struct tm *lptr; /* holds user login time */
120 static char *myname; /* pointer to invoker's name */
121 static char *mytty; /* holds device user is on */
122 static char nameval[sizeof (utmp->ut_user) + 1]; /* invoker's name */
123 static int number = 8; /* number of users per -q line */
124 static int optcnt = 0; /* keeps count of options */
125 static char outbuf[BUFSIZ]; /* buffer for output */
126 static char *program; /* holds name of this program */
127 #ifdef XPG4
128 static int aopt = 0; /* 1 = who -a */
129 static int dopt = 0; /* 1 = who -d */
130 #endif /* XPG4 */
131 static int qopt = 0; /* 1 = who -q */
132 static int sopt = 0; /* 1 = who -s */
133 static struct stat stbuf; /* area for stat buffer */
134 static struct stat *stbufp; /* ptr to structure */
135 static int terse = 1; /* 1 = print terse msgs */
136 static int Topt = 0; /* 1 = who -T */
137 static time_t timnow; /* holds current time */
138 static int totlusr = 0; /* cntr for users on system */
139 static int uopt = 0; /* 1 = who -u */
140 static char user[sizeof (utmp->ut_user) + 1]; /* holds user name */
141 static int validtype[UTMAXTYPE+1]; /* holds valid types */
142 static int wrap; /* flag to indicate wrap */
143 static char time_buf[128]; /* holds date and time string */
144 static char *end; /* used in strtol for end pointer */

146 int
147 main(int argc, char **argv)
148 {
149     int goerr = 0; /* non-zero indicates cmd error */
150     int i;
151     int optsw; /* switch for while of getopt() */

153     (void) setlocale(LC_ALL, "");

155 #if !defined(TEXT_DOMAIN) /* Should be defined by cc -D */
156 #define TEXT_DOMAIN "SYS_TEST" /* Use this only if it weren't */
157 #endif
158     (void) textdomain(TEXT_DOMAIN);

160     validtype[USER_PROCESS] = 1;
161     validtype[EMPTY] = 0;
162     stbufp = &stbuf;

164     /*
165      * Strip off path name of this command
166      */
167     for (i = strlen(argv[0]); i >= 0 && argv[0][i] != '/'; --i)
168         ;
169     for (i = strlen(argv[0]); i >= 0 && argv[0][i] != '/'; --i);
170     if (i >= 0)
171         argv[0] += i+1;
172     program = argv[0];

173     /*
174      * Buffer stdout for speed
175      */
176     setbuf(stdout, outbuf);

178     /*
179      * Retrieve options specified on command line
180      * XCU4 - add -m option

```

```

181     /*
182     while ((optsw = getopt(argc, argv, "abdHlmm:prstTu")) != EOF) {
183         optcnt++;
184         switch (optsw) {

186             case 'a':
187                 optcnt += 7;
188                 validtype[BOOT_TIME] = 1;
189                 validtype[DEAD_PROCESS] = 1;
190                 validtype[LOGIN_PROCESS] = 1;
191                 validtype[INIT_PROCESS] = 1;
192                 validtype[RUN_LVL] = 1;
193                 validtype[OLD_TIME] = 1;
194                 validtype[NEW_TIME] = 1;
195                 validtype[USER_PROCESS] = 1;
196 #ifdef XPG4
197                 aopt = 1;
198 #endif /* XPG4 */
199                 uopt = 1;
200                 Topt = 1;
201                 if (!sopt) terse = 0;
202                 break;

204             case 'b':
205                 validtype[BOOT_TIME] = 1;
206                 if (!uopt) validtype[USER_PROCESS] = 0;
207                 break;

209             case 'd':
210                 validtype[DEAD_PROCESS] = 1;
211                 if (!uopt) validtype[USER_PROCESS] = 0;
212 #ifdef XPG4
213                 dopt = 1;
214 #endif /* XPG4 */
215                 break;

217             case 'H':
218                 optcnt--; /* Don't count Header */
219                 Hopt = 1;
220                 break;

222             case 'l':
223                 validtype[LOGIN_PROCESS] = 1;
224                 if (!uopt) validtype[USER_PROCESS] = 0;
225                 terse = 0;
226                 break;

227             case 'm': /* New XCU4 option */
228                 justme = 1;
229                 break;

231             case 'n':
232                 errno = 0;
233                 number = strtol(optarg, &end, 10);
234                 if (errno != 0 || *end != '\0') {
235                     (void) fprintf(stderr, gettext(
236                         "%s: Invalid numeric argument\n"),
237                         program);
238                     exit(1);
239                 }
240                 if (number < 1) {
241                     (void) fprintf(stderr, gettext(
242                         "%s: Number of users per line must
243                         \"be at least 1\n\"), program);
244                     exit(1);
245                 }
246                 break;

```

```

248     case 'p':
249         validtype[INIT_PROCESS] = 1;
250         if (!uopt) validtype[USER_PROCESS] = 0;
251         break;

253     case 'q':
254         qopt = 1;
255         break;

257     case 'r':
258         validtype[RUN_LVL] = 1;
259         terse = 0;
260         if (!uopt) validtype[USER_PROCESS] = 0;
261         break;

263     case 's':
264         sopt = 1;
265         terse = 1;
266         break;

268     case 't':
269         validtype[OLD_TIME] = 1;
270         validtype[NEW_TIME] = 1;
271         if (!uopt) validtype[USER_PROCESS] = 0;
272         break;

274     case 'T':
275         Topt = 1;
276 #ifdef XPG4
277         terse = 1;      /* XPG4 requires -T */
278 #else /* XPG4 */
279         terse = 0;
280 #endif /* XPG4 */
281         break;

283     case 'u':
284         uopt = 1;
285         validtype[USER_PROCESS] = 1;
286         if (!sopt) terse = 0;
287         break;

289     case '?':
290         goerr++;
291         break;
292     default:
293         break;
294     }
295 }
296 #ifdef XPG4
297 /*
298  * XCU4 changes - check for illegal sopt, Topt & aopt combination
299  */
300 if (sopt == 1) {
301     terse = 1;
302     if (Topt == 1 || aopt == 1)
303         goerr++;
304 }
305 #endif /* XPG4 */

307 if (goerr > 0) {
308 #ifdef XPG4
309     /*
310     * XCU4 - slightly different usage with -s -a & -T
311     */
312     (void) fprintf(stderr, gettext("\nUsage:\t%s"), program);

```

```

313         (void) fprintf(stderr,
314             gettext("-s [-bdHlmpqrstu] [utmpx_like_file]\n"));

316     (void) fprintf(stderr, gettext(
317         "\t%s [-abdHlmpqrstu] [utmpx_like_file]\n"), program);
318 #else /* XPG4 */
319     (void) fprintf(stderr, gettext(
320         "\nUsage:\t%s [-abdHlmpqrstu] [utmpx_like_file]\n"),
321         program);
322 #endif /* XPG4 */
323     (void) fprintf(stderr,
324         gettext("\t%s -q [-n x] [utmpx_like_file]\n"), program);
325     (void) fprintf(stderr, gettext("\t%s [am i]\n"), program);
326     /*
327     * XCU4 changes - be explicit with "am i" options
328     */
329     (void) fprintf(stderr, gettext("\t%s [am I]\n"), program);
330     (void) fprintf(stderr, gettext(
331         "a\tall (bdlprrtu options)\n"));
332     (void) fprintf(stderr, gettext("b\tboot time\n"));
333     (void) fprintf(stderr, gettext("d\tdead processes\n"));
334     (void) fprintf(stderr, gettext("H\tprint header\n"));
335     (void) fprintf(stderr, gettext("l\tlogin processes\n"));
336     (void) fprintf(stderr, gettext(
337         "n #\tspecify number of users per line for -q\n"));
338     (void) fprintf(stderr,
339         gettext("p\tprocesses other than getty or users\n"));
340     (void) fprintf(stderr, gettext("q\tquick %s\n"), program);
341     (void) fprintf(stderr, gettext("r\ttrun level\n"));
342     (void) fprintf(stderr, gettext(
343         "s\tshort form of %s (no time since last output or pid)\n"),
344         program);
345     (void) fprintf(stderr, gettext("t\ttime changes\n"));
346     (void) fprintf(stderr, gettext(
347         "T\tstatus of tty (+ writable, - not writable, "
348         "? hung)\n"));
349     (void) fprintf(stderr, gettext("u\tuseful information\n"));
350     (void) fprintf(stderr,
351         gettext("m\tinformation only about current terminal\n"));
352     (void) fprintf(stderr, gettext(
353         "am i\tinformation about current terminal "
354         "(same as -m)\n"));
355     (void) fprintf(stderr, gettext(
356         "am I\tinformation about current terminal "
357         "(same as -m)\n"));
358     exit(1);
359 }

361 /*
362  * XCU4: If -q option ignore all other options
363  */
364 if (qopt == 1) {
365     Hopt = 0;
366     sopt = 0;
367     Topt = 0;
368     uopt = 0;
369     justme = 0;
370     validtype[ACCOUNTING] = 0;
371     validtype[BOOT_TIME] = 0;
372     validtype[DEAD_PROCESS] = 0;
373     validtype[LOGIN_PROCESS] = 0;
374     validtype[INIT_PROCESS] = 0;
375     validtype[RUN_LVL] = 0;
376     validtype[OLD_TIME] = 0;
377     validtype[NEW_TIME] = 0;
378     validtype[USER_PROCESS] = 1;

```

```

379     }
381     if (argc == optind + 1) {
382         optcnt++;
383         ck_file(argv[optind]);
384         (void) utmpxname(argv[optind]);
385     }
387     /*
388     *   Test for 'who am i' or 'who am I'
389     *   XCU4 - check if justme was already set by -m option
390     */
391     if (justme == 1 || (argc == 3 && strcmp(argv[1], "am") == 0 &&
392         ((argv[2][0] == 'i' || argv[2][0] == 'I') &&
393         argv[2][1] == '\0'))) {
394         justme = 1;
395         myname = nameval;
396         (void) cuserid(myname);
397         if ((mytty = ttyname(fileno(stdin))) == NULL &&
398             (mytty = ttyname(fileno(stdout))) == NULL &&
399             (mytty = ttyname(fileno(stderr))) == NULL) {
400             (void) fprintf(stderr, gettext(
401                 "Must be attached to terminal for 'am I' option\n"));
402             (void) fflush(stderr);
403             exit(1);
404         } else
405             mytty += 5; /* bump past "/dev/" */
406     }
408     if (!terse) {
409         if (Hopt)
410             (void) printf(gettext(
411                 "NAME      LINE      TIME      IDLE      PID      COMMENTS\n"));
413         timnow = time(0);
415         if ((fildes = open("/etc/inittab",
416             O_NONBLOCK|O_RDONLY)) == -1) {
417             (void) snprintf(errmsg, sizeof (errmsg),
418                 gettext("%s: Cannot open /etc/inittab"), program);
419             perror(errmsg);
420             exit(errno);
421         }
423         if (fstat(fildes, stbufp) == -1) {
424             (void) snprintf(errmsg, sizeof (errmsg),
425                 gettext("%s: Cannot stat /etc/inittab"), program);
426             perror(errmsg);
427             exit(errno);
428         }
430         if ((inittab = malloc(stbufp->st_size + 1)) == NULL) {
431             (void) snprintf(errmsg, sizeof (errmsg),
432                 gettext("%s: Cannot allocate %ld bytes"),
433                 program, stbufp->st_size);
434             perror(errmsg);
435             exit(errno);
436         }
438         if (read(fildes, inittab, stbufp->st_size)
439             != stbufp->st_size) {
440             (void) snprintf(errmsg, sizeof (errmsg),
441                 gettext("%s: Error reading /etc/inittab"),
442                 program);
443             perror(errmsg);
444             exit(errno);

```

```

445     }
447         inittab[stbufp->st_size] = '\0';
448         iinit = inittab;
449     } else {
450         if (Hopt) {
451             #ifdef XPG4
452                 if (dopt) {
453                     (void) printf(gettext(
454                         "NAME      LINE      TIME      COMMENTS\n"));
455                 } else {
456                     (void) printf(
457                         gettext("NAME      LINE      TIME\n"));
458                 }
459             #else /* XPG4 */
460                 (void) printf(
461                     gettext("NAME      LINE      TIME\n"));
462             #endif /* XPG4 */
463         }
464     }
465     process();
467     /*
468     *   'who -q' requires EOL upon exit,
469     *   followed by total line
470     */
471     if (qopt)
472         (void) printf(gettext("\n# users=%d\n"), totlusrs);
473     return (0);
474 }
476 static void
477 dump()
478 {
479     char    device[sizeof (utmp->ut_line) + 1];
480     time_t  hr;
481     time_t  idle;
482     time_t  min;
483     char    path[sizeof (utmp->ut_line) + 6];
484     int     pexit;
485     int     pterm;
486     int     rc;
487     char    w; /* writeability indicator */
489     /*
490     * Get and check user name
491     */
492     if (utmp->ut_user[0] == '\0')
493         (void) strcpy(user, " .");
494     else {
495         (void) strncpy(user, utmp->ut_user, sizeof (user));
496         user[sizeof (user) - 1] = '\0';
497     }
498     totlusrs++;
500     /*
501     * Do print in 'who -q' format
502     */
503     if (qopt) {
504         /*
505         * XCU4 - Use non user macro for correct user count
506         */
507         if (((totlusrs - 1) % number) == 0 && totlusrs > 1)
508             (void) printf("\n");
509         (void) printf("%-*. *s ", LOGIN_WIDTH, NMAX, user);
510         (void) printf("%-*s ", NMAX, user);

```

```

510         return;
511     }

514     pexit = (int)' ';
515     pterm = (int)' ';

517     /*
518     *     Get exit info if applicable
519     */
520     if (utmpp->ut_type == RUN_LVL || utmpp->ut_type == DEAD_PROCESS) {
521         pterm = utmpp->ut_exit.e_termination;
522         pexit = utmpp->ut_exit.e_exit;
523     }

525     /*
526     *     Message ut_xtime field
527     */
528     lptr = localtime(&utmpp->ut_xtime);
529     (void) strftime(time_buf, sizeof (time_buf),
530         dcgettext(NULL, DATE_FMT, LC_TIME), lptr);

532     /*
533     *     Get and message device
534     */
535     if (utmpp->ut_line[0] == '\0')
536         (void) strcpy(device, " .");
537     else {
538         (void) strncpy(device, utmpp->ut_line,
539             sizeof (utmpp->ut_line));
540         device[sizeof (utmpp->ut_line)] = '\0';
541     }

543     /*
544     *     Get writeability if requested
545     *     XCU4 - only print + or - for user processes
546     */
547     if (Topt && (utmpp->ut_type == USER_PROCESS)) {
548         w = '-';
549         (void) strcpy(path, "/dev/");
550         (void) strncpy(path + 5, utmpp->ut_line,
551             sizeof (utmpp->ut_line));
552         path[5 + sizeof (utmpp->ut_line)] = '\0';

554         if ((rc = stat(path, stbufp)) == -1) w = '?';
555         else if ((stbufp->st_mode & S_IWOTH) ||
556             (stbufp->st_mode & S_IWGRP)) /* Check group & other */
557             w = '+';

559     } else
560         w = ' ';

562     /*
563     *     Print the TERSE portion of the output
564     */
565     (void) printf("%-*.s %c %-12s %s", LOGIN_WIDTH, NMAX, user,
566         w, device, time_buf);
568     (void) printf("%-*.s %c %-12s %s", NMAX, user, w, device, time_buf);

568     if (!terse) {
569         /*
570         *     Stat device for idle time
571         *     (Don't complain if you can't)
572         */
573         rc = -1;
574         if (utmpp->ut_type == USER_PROCESS) {

```

```

575         (void) strcpy(path, "/dev/");
576         (void) strncpy(path + 5, utmpp->ut_line,
577             sizeof (utmpp->ut_line));
578         path[5 + sizeof (utmpp->ut_line)] = '\0';
579         rc = stat(path, stbufp);
580     }
581     if (rc != -1) {
582         idle = timnow - stbufp->st_mtime;
583         hr = idle/3600;
584         min = (unsigned)(idle/60)%60;
585         if (hr == 0 && min == 0)
586             (void) printf(gettext(" . "));
587         else {
588             if (hr < 24)
589                 (void) printf("%2d:%2.2d", (int)hr,
590                     (int)min);
591             else
592                 (void) printf(gettext(" old "));
593         }
594     }

596     /*
597     *     Add PID for verbose output
598     */
599     if (utmpp->ut_type != BOOT_TIME &&
600         utmpp->ut_type != RUN_LVL &&
601         utmpp->ut_type != ACCOUNTING)
602         (void) printf(" %5ld", utmpp->ut_pid);

604     /*
605     *     Handle /etc/inittab comment
606     */
607     if (utmpp->ut_type == DEAD_PROCESS) {
608         (void) printf(gettext(" id=%4.4s "),
609             utmpp->ut_id);
610         (void) printf(gettext("term=%-3d "), pterm);
611         (void) printf(gettext("exit=%d "), pexit);
612     } else if (utmpp->ut_type != INIT_PROCESS) {
613         /*
614         *     Search for each entry in inittab
615         *     string. Keep our place from
616         *     search to search to try and
617         *     minimize the work. Wrap once if needed
618         *     for each entry.
619         */
620         wrap = 0;
621         /*
622         *     Look for a line beginning with
623         *     utmpp->ut_id
624         */
625         while ((rc = strncmp(utmpp->ut_id, iinit,
626             strcspn(iinit, ":")) != 0) {
627             for (; *iinit != '\n'; iinit++)
628                 ;
629             for (; *iinit != '\n'; iinit++);
629             iinit++;

631         /*
632         *     Wrap once if necessary to
633         *     find entry in inittab
634         */
635         if (*iinit == '\0') {
636             if (!wrap) {
637                 iinit = inittab;
638                 wrap = 1;
639             }

```

```

640     }
641     }
642
643     if (*iinit != '\0') {
644         /*
645          *       We found our entry
646          */
647         for (iinit++; *iinit != '#' &&
648              *iinit != '\n'; iinit++)
649             ;
650         if (*iinit == '#') {
651             for (iinit++; *iinit == ' ' ||
652                  *iinit == '\t'; iinit++)
653                 ;
654             for (rc = 0; *iinit != '\n'; iinit++)
655                 comment[rc++] = *iinit;
656             comment[rc] = '\0';
657         } else
658             (void) strcpy(comment, " ");
659
660         (void) printf(" %s", comment);
661     } else
662         iinit = inittab; /* Reset pointer */
663 }
664 if (utmp->ut_type == INIT_PROCESS)
665     (void) printf(gettext(" id=%4.4s"), utmp->ut_id);
666 }
667 #ifdef XPG4
668 else
669     if (dopt && utmp->ut_type == DEAD_PROCESS) {
670         (void) printf(gettext("\tterm=%-3d "), pterm);
671         (void) printf(gettext("exit=%d "), pexit);
672     }
673 #endif /* XPG4 */
674
675 /*
676 *       Handle RUN_LVL process - If no alt. file - Only one!
677 */
678 if (utmp->ut_type == RUN_LVL) {
679     (void) printf(" %c %5ld %c", pterm, utmp->ut_pid,
680                 pexit);
681     if (optcnt == 1 && !validtype[USER_PROCESS]) {
682         (void) printf("\n");
683         exit(0);
684     }
685 }
686
687 /*
688 *       Handle BOOT_TIME process - If no alt. file - Only one!
689 */
690 if (utmp->ut_type == BOOT_TIME) {
691     if (optcnt == 1 && !validtype[USER_PROCESS]) {
692         (void) printf("\n");
693         exit(0);
694     }
695 }
696
697 /*
698 *       Get remote host from utmp structure
699 */
700 if (utmp && utmp->ut_host[0])
701     (void) printf("\t(%.*s)", sizeof (utmp->ut_host),
702                 utmp->ut_host);

```

```

705     /*
706     *       Now, put on the trailing EOL
707     */
708     (void) printf("\n");
709 }
710
711 static void
712 process()
713 {
714     struct passwd *pwp;
715     int i = 0;
716     char *ttname;
717
718     /*
719     *       Loop over each entry in /var/adm/utmpx
720     */
721
722     setutxent();
723     while ((utmp = getutxent()) != NULL) {
724 #ifdef DEBUG
725         (void) printf(
726             "ut_user '%s'\nut_id '%s'\nut_line '%s'\nut_type '%d'\n\n",
727             utmp->ut_user, utmp->ut_id, utmp->ut_line, utmp->ut_type);
728 #endif
729         if (utmp->ut_type <= UTMAXTYPE) {
730             /*
731              *       Handle "am i"
732              */
733             if (justme) {
734                 if (strncmp(myname, utmp->ut_user,
735                             sizeof (utmp->ut_user)) == 0 &&
736                     strncmp(mytty, utmp->ut_line,
737                             sizeof (utmp->ut_line)) == 0 &&
738                     utmp->ut_type == USER_PROCESS) {
739                     /*
740                      * we have have found ourselves
741                      * in the utmp file and the entry
742                      * is a user process, this is not
743                      * meaningful otherwise
744                      */
745
746                     dump();
747                     exit(0);
748                 }
749                 continue;
750             }
751
752             /*
753              *       Print the line if we want it
754              */
755             if (validtype[utmp->ut_type]) {
756 #ifdef XPG4
757                 if (utmp->ut_type == LOGIN_PROCESS) {
758                     if ((utmp->ut_line[0] == '\0') ||
759                         (strcmp(utmp->ut_user,
760                                "LOGIN") != 0))
761                         (strcmp(utmp->ut_user, "LOGIN") != 0))
762                             continue;
763                 }
764 #endif /* XPG4 */
765                 dump();
766             }
767         } else {
768             (void) fprintf(stderr,

```

```
769         gettext("%s: Error --- entry has ut_type "  
770         "of %d\n"), program, utmpp->ut_type);  
771         (void) fprintf(stderr,  
772         gettext(" when maximum is %d\n"), UTMATYPE);  
773     }  
774 }  
  
776 /*  
777  * If justme is set at this point than the utmp entry  
778  * was not found.  
779  */  
780 if (justme) {  
781     static struct utmpx utmpt;  
  
783     pwp = getpwuid(geteuid());  
784     if (pwp != NULL)  
785         while (i < (int)sizeof (utmpt.ut_user) &&  
786                *pwp->pw_name != 0)  
787             utmpt.ut_user[i++] = *pwp->pw_name++;  
  
789     ttname = ttyname(1);  
  
791     i = 0;  
792     if (ttname != NULL)  
793         while (i < (int)sizeof (utmpt.ut_line) &&  
794                *ttname != 0)  
795             utmpt.ut_line[i++] = *ttname++;  
  
797     utmpt.ut_id[0] = 0;  
798     utmpt.ut_pid = getpid();  
799     utmpt.ut_type = USER_PROCESS;  
800     (void) time(&utmpt.ut_xtime);  
801     utmpp = &utmpt;  
802     dump();  
803     exit(0);  
804 }  
805 }  
  
_____unchanged_portion_omitted_____
```

new/usr/src/cmd/whodo/whodo.c

1

```
*****
21012 Sat Jun 1 10:02:50 2013
new/usr/src/cmd/whodo/whodo.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
24  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
25  * Use is subject to license terms.
26  */

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

31 /*
32  * University Copyright- Copyright (c) 1982, 1986, 1988
33  * The Regents of the University of California
34  * All Rights Reserved
35  *
36  * University Acknowledgment- Portions of this document are derived from
37  * software developed by the University of California, Berkeley, and its
38  * contributors.
39  */

41 /*
42  * This is the new whodo command which takes advantage of
43  * the /proc interface to gain access to the information
44  * of all the processes currently on the system.
45  *
46  * Maintenance note:
47  *
48  * Much of this code is replicated in w.c. If you're
49  * fixing bugs here, then you should probably fix 'em there too.
50  */

52 #include <stdio.h>
53 #include <string.h>
54 #include <stdlib.h>
55 #include <ctype.h>
56 #include <fcntl.h>
57 #include <time.h>
58 #include <errno.h>
59 #include <sys/types.h>
60 #include <utmpx.h>
```

new/usr/src/cmd/whodo/whodo.c

2

```
61 #include <sys/utsname.h>
62 #include <sys/stat.h>
63 #include <sys/mkdev.h>
64 #include <dirent.h>
65 #include <procfs.h> /* /proc header file */
66 #include <sys/wait.h>
67 #include <locale.h>
68 #include <unistd.h>
69 #include <limits.h>
70 #include <priv_utils.h>

72 /*
73  * Use the full lengths from utmpx for user and line.
74  * utmpx defines wider fields for user and line. For compatibility of output,
75  * we are limiting these to the old maximums in utmp. Define UTMPX_NAMELEN
76  * to use the full lengths.
77  */
78 #define NMAX (sizeof (((struct utmpx *)0)->ut_user))
79 #define LMAX (sizeof (((struct utmpx *)0)->ut_line))
80 #ifndef UTMPX_NAMELEN
81 /* XXX - utmp - fix name length */
82 #define NMAX (_POSIX_LOGIN_NAME_MAX - 1)
83 #define LMAX 12
84 #else /* UTMPX_NAMELEN */
85 static struct utmpx dummy;
86 #define NMAX (sizeof (dummy.ut_user))
87 #define LMAX (sizeof (dummy.ut_line))
88 #endif /* UTMPX_NAMELEN */

89 /* Print minimum field widths. */
90 #define LOGIN_WIDTH 8
91 #define LINE_WIDTH 12

92 #define DIV60(t) ((t+30)/60) /* x/60 rounded */

93 #ifdef ERR
94 #undef ERR
95 #endif
96 #define ERR (-1)

97 #define DEVNAMELEN 14
98 #define HSIZE 256 /* size of process hash table */
99 #define PROCDIR "/proc"
100 #define INITPROCESS (pid_t)1 /* init process pid */
101 #define NONE 'n' /* no state */
102 #define RUNNING 'r' /* runnable process */
103 #define ZOMBIE 'z' /* zombie process */
104 #define VISITED 'v' /* marked node as visited */

105 static int ndevs; /* number of configured devices */
106 static int maxdev; /* slots for configured devices */
107 #define DNINCR 100
108 static struct devl { /* device list */
109     char dname[DEVNAMELEN]; /* device name */
110     dev_t ddev; /* device number */
111 } *devl;

112 #ifndef unchanged_portion_omitted
113 /*
114  * define hash table for struct uproc
115  * Hash function uses process id
116  * and the size of the hash table(HSIZE)
117  * to determine process index into the table.
118  */
119 static struct uproc pr_htbl[HSIZE];
120 */
121 #endif
```

```

129 static struct uproc *findhash(pid_t);
130 static time_t findidle(char *);
131 static void clnarglist(char *);
132 static void showproc(struct uproc *);
133 static void showtotals(struct uproc *);
134 static void calctotals(struct uproc *);
135 static char *getty(dev_t);
136 static void prttime(time_t, char *);
137 static void prtat(time_t *);
138 static void checkampm(char *);

140 static char *prog;
141 static int header = 1; /* true if -h flag: don't print heading */
142 static int lflag = 0; /* true if -l flag: w command format */
143 static char *sel_user; /* login of particular user selected */
144 static time_t now; /* current time of day */
145 static time_t uptime; /* time of last reboot & elapsed time since */
146 static int nusers; /* number of users logged in now */
147 static time_t idle; /* number of minutes user is idle */
148 static time_t jobtime; /* total cpu time visible */
149 static char doing[520]; /* process attached to terminal */
150 static time_t proctime; /* cpu time of process in doing */
151 static int empty;
152 static pid_t curpid;

154 #if SIGQUIT > SIGINT
155 #define ACTSIZE SIGQUIT
156 #else
157 #define ACTSIZE SIGINT
158 #endif

160 int
161 main(int argc, char *argv[])
162 {
163     struct utmpx *ut;
164     struct utmpx *utmpbegin;
165     struct utmpx *utmpend;
166     struct utmpx *utp;
167     struct tm *tm;
168     struct uproc *up, *parent, *pgrp;
169     struct psinfo info;
170     struct sigaction actinfo[ACTSIZE];
171     struct pstatus statinfo;
172     size_t size;
173     struct stat sbuf;
174     struct utsname uts;
175     DIR *dirp;
176     struct dirent *dp;
177     char pname[64];
178     char *fname;
179     int procfid;
180     int i;
181     int days, hrs, mins;
182     int entries;

184     /*
185      * This program needs the proc_owner privilege
186      */
187     (void) __init_suid_priv(PU_CLEARLIMITSET, PRIV_PROC_OWNER,
188                          (char *)NULL);

190     (void) setlocale(LC_ALL, "");
191 #if !defined(TEXT_DOMAIN)
192 #define TEXT_DOMAIN "SYS_TEST"
193 #endif
194     (void) textdomain(TEXT_DOMAIN);

```

```

196     prog = argv[0];

198     while (argc > 1) {
199         if (argv[1][0] == '-') {
200             for (i = 1; argv[1][i]; i++) {
201                 switch (argv[1][i]) {

203                     case 'h':
204                         header = 0;
205                         break;

207                     case 'l':
208                         lflag++;
209                         break;

211                     default:
212                         (void) printf(gettext(
213                             "usage: %s [ -hl ] [ user ]\n"),
214                             prog);
215                         exit(1);
216                 }
217             }
218         } else {
219             if (!isalnum(argv[1][0]) || argc > 2) {
220                 (void) printf(gettext(
221                     "usage: %s [ -hl ] [ user ]\n"), prog);
222                 exit(1);
223             } else
224                 sel_user = argv[1];
225         }
226         argc--; argv++;
227     }

229     /*
230      * read the UTMPX_FILE (contains information about
231      * each logged in user)
232      */
233     if (stat(UTMPX_FILE, &sbuf) == ERR) {
234         (void) fprintf(stderr, gettext("%s: stat error of %s: %s\n"),
235                        prog, UTMPX_FILE, strerror(errno));
236         exit(1);
237     }
238     entries = sbuf.st_size / sizeof (struct futmpx);
239     size = sizeof (struct utmpx) * entries;

241     if ((ut = malloc(size)) == NULL) {
242         (void) fprintf(stderr, gettext("%s: malloc error of %s: %s\n"),
243                        prog, UTMPX_FILE, strerror(errno));
244         exit(1);
245     }

247     (void) utmpxname(UTMPX_FILE);

249     utmpbegin = ut;
250     /* LINTED pointer cast may result in improper alignment */
251     utmpend = (struct utmpx *)((char *)utmpbegin + size);

253     setutxent();
254     while ((ut < utmpend) && ((utp = getutxent()) != NULL))
255         (void) memcpy(ut++, utp, sizeof (*ut));
256     endutxent();

258     (void) time(&now); /* get current time */

260     if (header) { /* print a header */

```



```

261     if (lflag) { /* w command format header */
262         prtat(&now);
263         for (ut = utmpbegin; ut < utmpend; ut++) {
264             if (ut->ut_type == USER_PROCESS) {
265                 nusers++;
266             } else if (ut->ut_type == BOOT_TIME) {
267                 uptime = now - ut->ut_xtime;
268                 uptime += 30;
269                 days = uptime / (60*60*24);
270                 uptime %= (60*60*24);
271                 hrs = uptime / (60*60);
272                 uptime %= (60*60);
273                 mins = uptime / 60;
274
275                 (void) printf(dcgettext(NULL,
276                     " up %d day(s), %d hr(s), "
277                     "%d min(s)", LC_TIME),
278                     days, hrs, mins);
279             }
280         }
281
282         ut = utmpbegin; /* rewind utmp data */
283         (void) printf(dcgettext(NULL,
284             " %d user(s)\n", LC_TIME), nusers);
285         (void) printf(dcgettext(NULL, "User      tty
286             "login@ idle JCPU PCPU what\n", LC_TIME));
287     } else { /* standard whodo header */
288         char date_buf[100];
289
290         /*
291          * print current time and date
292          */
293         (void) strftime(date_buf, sizeof (date_buf),
294             dcgettext(NULL, "%C", LC_TIME), localtime(&now));
295         (void) printf("%s\n", date_buf);
296
297         /*
298          * print system name
299          */
300         (void) uname(&uts);
301         (void) printf("%s\n", uts.nodename);
302     }
303 }
304
305 /*
306 * loop through /proc, reading info about each process
307 * and build the parent/child tree
308 */
309 if (!(dirp = opendir(PROCDIR))) {
310     (void) fprintf(stderr, gettext("%s: could not open %s: %s\n"),
311         prog, PROCDIR, strerror(errno));
312     exit(1);
313 }
314
315 while ((dp = readdir(dirp)) != NULL) {
316     if (dp->d_name[0] == '.')
317         continue;
318 retry:
319     (void) snprintf(pname, sizeof (pname),
320         "%s/%s/", PROCDIR, dp->d_name);
321     fname = pname + strlen(pname);
322     (void) strcpy(fname, "psinfo");
323     if ((procfd = open(pname, O_RDONLY)) < 0)
324         continue;
325     if (read(procfd, &info, sizeof (info)) != sizeof (info)) {
326         int err = errno;

```

```

327         (void) close(procfd);
328         if (err == EAGAIN)
329             goto retry;
330         if (err != ENOENT)
331             (void) fprintf(stderr, gettext(
332                 "%s: read() failed on %s: %s\n"),
333                 prog, pname, strerror(err));
334         continue;
335     }
336     (void) close(procfd);
337
338     up = findhash(info.pr_pid);
339     up->p_ttyd = info.pr_ttydev;
340     up->p_state = (info.pr_nlwp == 0? ZOMBIE : RUNNING);
341     up->p_time = 0;
342     up->p_ctime = 0;
343     up->p_igintr = 0;
344     (void) strncpy(up->p_comm, info.pr_fname,
345         sizeof (info.pr_fname));
346     up->p_args[0] = 0;
347
348     if (up->p_state != NONE && up->p_state != ZOMBIE) {
349         (void) strcpy(fname, "status");
350
351         /* now we need the proc_owner privilege */
352         (void) __priv_bracket(PRIV_ON);
353
354         procfd = open(pname, O_RDONLY);
355
356         /* drop proc_owner privilege after open */
357         (void) __priv_bracket(PRIV_OFF);
358
359         if (procfd < 0)
360             continue;
361
362         if (read(procfd, &stainfo, sizeof (stainfo))
363             != sizeof (stainfo)) {
364             int err = errno;
365             (void) close(procfd);
366             if (err == EAGAIN)
367                 goto retry;
368             if (err != ENOENT)
369                 (void) fprintf(stderr, gettext(
370                     "%s: read() failed on %s: %s\n"),
371                     prog, pname, strerror(err));
372             continue;
373         }
374         (void) close(procfd);
375
376         up->p_time = stainfo.pr_utime.tv_sec +
377             stainfo.pr_stime.tv_sec;
378         up->p_ctime = stainfo.pr_cutime.tv_sec +
379             stainfo.pr_cstime.tv_sec;
380
381         (void) strcpy(fname, "sigact");
382
383         /* now we need the proc_owner privilege */
384         (void) __priv_bracket(PRIV_ON);
385
386         procfd = open(pname, O_RDONLY);
387
388         /* drop proc_owner privilege after open */
389         (void) __priv_bracket(PRIV_OFF);
390
391         if (procfd < 0)
392             continue;

```

```

393     if (read(procf, actinfo, sizeof (actinfo))
394         != sizeof (actinfo)) {
395         int err = errno;
396         (void) close(procf);
397         if (err == EAGAIN)
398             goto retry;
399         if (err != ENOENT)
400             (void) fprintf(stderr, gettext(
401                 "%s: read() failed on %s: %s\n"),
402                 prog, pname, strerror(err));
403         continue;
404     }
405     (void) close(procf);
407     up->p_igintr =
408         actinfo[SIGINT-1].sa_handler == SIG_IGN &&
409         actinfo[SIGQUIT-1].sa_handler == SIG_IGN;
411     up->p_args[0] = 0;
413     /*
414      * Process args if there's a chance we'll print it.
415      */
416     if (lflag) { /* w command needs args */
417         clnarglist(info.pr_psargs);
418         (void) strcpy(up->p_args, info.pr_psargs);
419         if (up->p_args[0] == 0 ||
420             up->p_args[0] == '-' &&
421             up->p_args[1] <= ' ' ||
422             up->p_args[0] == '?' ) {
423             (void) strcat(up->p_args, " (");
424             (void) strcat(up->p_args, up->p_comm);
425             (void) strcat(up->p_args, ")");
426         }
427     }
429 }
431 /*
432  * link pgrp together in case parents go away
433  * Pgrp chain is a single linked list originating
434  * from the pgrp leader to its group member.
435  */
436 if (info.pr_pgid != info.pr_pid) { /* not pgrp leader */
437     pgrp = findhash(info.pr_pgid);
438     up->p_pgrpplink = pgrp->p_pgrpplink;
439     pgrp->p_pgrpplink = up;
440 }
441 parent = findhash(info.pr_ppid);
443 /* if this is the new member, link it in */
444 if (parent->p_upid != INITPROCESS) {
445     if (parent->p_child) {
446         up->p_sibling = parent->p_child;
447         up->p_child = 0;
448     }
449     parent->p_child = up;
450 }
452 }
454 /* revert to non-privileged user */
455 (void) __priv_relinquish();
457 (void) closedir(dirp);
458 (void) time(&now); /* get current time */

```

```

460     /*
461      * loop through utmpx file, printing process info
462      * about each logged in user
463      */
464     for (ut = utmpbegin; ut < utmpend; ut++) {
465         time_t tim;
467         if (ut->ut_type != USER_PROCESS)
468             continue;
469         if (sel_user && strcmp(ut->ut_name, sel_user, NMAX) != 0)
470             continue; /* we're looking for somebody else */
471         if (lflag) { /* -l flag format (w command) */
472             /* print login name of the user */
473             (void) printf("%-*.s ", LOGIN_WIDTH, (int)NMAX,
474                 ut->ut_name);
475             (void) printf("%-*.s ", NMAX, NMAX, ut->ut_name);
476
477             /* print tty user is on */
478             (void) printf("%-*.s", LINE_WIDTH, (int)LMAX,
479                 ut->ut_line);
480             (void) printf("%-*.s", LMAX, LMAX, ut->ut_line);
481
482             /* print when the user logged in */
483             tim = ut->ut_xtime;
484             (void) prtcat(&tim);
485
486             /* print idle time */
487             idle = findidle(ut->ut_line);
488             if (idle >= 36 * 60)
489                 (void) printf(dcgettext(NULL, "%2ddays ",
490                     LC_TIME), (idle + 12 * 60) / (24 * 60));
491             else
492                 prttime(idle, " ");
493             showtotals(findhash((pid_t)ut->ut_pid));
494         } else { /* standard whodo format */
495             tim = ut->ut_xtime;
496             tm = localtime(&tim);
497             (void) printf("\n%-*.s %-*.s %2.1d:%2.2d\n",
498                 LINE_WIDTH, (int)LMAX, ut->ut_line,
499                 LOGIN_WIDTH, (int)NMAX, ut->ut_name, tm->tm_hour,
500                 tm->tm_min);
501             (void) printf("%-*.s %-*.s %2.1d:%2.2d\n",
502                 LMAX, LMAX, ut->ut_line,
503                 NMAX, NMAX, ut->ut_name, tm->tm_hour, tm->tm_min);
504             showproc(findhash((pid_t)ut->ut_pid));
505         }
506     }
507     return (0);
508 }
509 /*
510  * Used for standard whodo format.
511  * This is the recursive routine descending the process
512  * tree starting from the given process pointer(up).
513  * It used depth-first search strategy and also marked
514  * each node as printed as it traversed down the tree.
515  */
516 static void
517 showproc(struct uproc *up)
518 {
519     struct uproc *zp;
520
521     if (up->p_state == VISITED) /* we already been here */
522         return;
523     /* print the data for this process */

```

```
521     if (up->p_state == ZOMBIE)
522         (void) printf("    %-*.*s %5d %4.1ld:%2.2ld %s\n",
523             LINE_WIDTH, (int)LMAX, "  ?", (int)up->p_upid, 0L, 0L,
524             "<defunct>");
525     else if (up->p_state != NONE) {
526         (void) printf("    %-*.*s %5d %4.1ld:%2.2ld %s\n",
527             LINE_WIDTH, (int)LMAX, getty(up->p_ttyd), (int)up->p_upid,
528             LMAX, LMAX, getty(up->p_ttyd), (int)up->p_upid,
529             up->p_time / 60L, up->p_time % 60L,
530             up->p_comm);
531     }
532     up->p_state = VISITED;
533
534     /* descend for its children */
535     if (up->p_child) {
536         showproc(up->p_child);
537         for (zp = up->p_child->p_sibling; zp; zp = zp->p_sibling) {
538             showproc(zp);
539         }
540     }
541
542     /* print the pgrp relation */
543     if (up->p_pgrplink)
544         showproc(up->p_pgrplink);
545 }
546
547 _____
548 unchanged_portion_omitted
```

new/usr/src/head/limits.h

1

```
*****
10785 Sat Jun 1 10:02:50 2013
new/usr/src/head/limits.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
25  * Copyright 2008 Sun Microsystems, Inc. All rights reserved.
26  * Use is subject to license terms.
27  */

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

33 #ifndef _LIMITS_H
34 #define _LIMITS_H

34 #pragma ident "%Z%M% %I% %E% SMI" /* SVr4.0 1.34 */

36 #include <sys/feature_tests.h>
37 #include <sys/isa_defs.h>
38 #include <iso/limits_iso.h>

40 /*
41  * Include fixed width type limits as proposed by the ISO/JTC1/SC22/WG14 C
42  * committee's working draft for the revision of the current ISO C standard,
43  * ISO/IEC 9899:1990 Programming language - C. These are not currently
44  * required by any standard but constitute a useful, general purpose set
45  * of type definitions and limits which is namespace clean with respect to
46  * all standards.
47  */
48 #if defined(__EXTENSIONS__) || !defined(_STRICT_STDC) || \
49     defined(__XOPEN_OR_POSIX)
50 #include <sys/int_limits.h>
51 #endif

53 #ifdef __cplusplus
54 extern "C" {
55 #endif

57 #if defined(__EXTENSIONS__) || !defined(_STRICT_STDC) || \
58     defined(__XOPEN_OR_POSIX)
```

new/usr/src/head/limits.h

2

```
60 #define SSIZE_MAX LONG_MAX /* max value of an "ssize_t" */

62 /*
63  * ARG_MAX is calculated as follows:
64  * NCARGS - space for other stuff on initial stack
65  * like aux vectors, saved registers, etc..
66  */
67 #define _ARG_MAX32 1048320 /* max length of args to exec 32-bit program */
68 #define _ARG_MAX64 2096640 /* max length of args to exec 64-bit program */
69 #ifdef _LP64
70 #define ARG_MAX _ARG_MAX64 /* max length of arguments to exec */
71 #else /* _LP64 */
72 #define ARG_MAX _ARG_MAX32 /* max length of arguments to exec */
73 #endif /* _LP64 */

75 #ifndef MAX_CANON
76 #define MAX_CANON 256 /* max bytes in line for canonical processing */
77 #endif

79 #ifndef MAX_INPUT
80 #define MAX_INPUT 512 /* max size of a char input buffer */
81 #endif

83 #define NGROUPS_MAX 16 /* max number of groups for a user */

85 #ifndef PATH_MAX
86 #define PATH_MAX 1024 /* max # of characters in a path name */
87 #endif

89 #define SYMLINK_MAX 1024 /* max # of characters a symlink can contain */

91 #define PIPE_BUF 5120 /* max # bytes atomic in write to a pipe */

93 #ifndef TMP_MAX
94 #define TMP_MAX 17576 /* 26 * 26 * 26 */
95 #endif

97 /*
98  * POSIX conformant definitions - An implementation may define
99  * other symbols which reflect the actual implementation. Alternate
100 * definitions may not be as restrictive as the POSIX definitions.
101 */
102 #define _POSIX_AIO_LISTIO_MAX 2
103 #define _POSIX_AIO_MAX 1
104 #define _POSIX_ARG_MAX 4096
105 #ifdef _XPG6
106 #define _POSIX_CHILD_MAX 25
107 #else
108 #define _POSIX_CHILD_MAX 6 /* POSIX.1-1990 default */
109 #endif
110 #define _POSIX_CLOCKRES_MIN 20000000
111 #define _POSIX_DELAYTIMER_MAX 32
112 #define _POSIX_LINK_MAX 8
113 #define _POSIX_MAX_CANON 255
114 #define _POSIX_MAX_INPUT 255
115 #define _POSIX_MQ_OPEN_MAX 8
116 #define _POSIX_MQ_PRIO_MAX 32
117 #define _POSIX_NAME_MAX 14
118 #ifdef _XPG6
119 #define _POSIX_NGROUPS_MAX 8
120 #define _POSIX_OPEN_MAX 20
121 #define _POSIX_PATH_MAX 256
122 #else
123 #define _POSIX_NGROUPS_MAX 0 /* POSIX.1-1990 defaults */
124 #define _POSIX_OPEN_MAX 16
```

```

125 #define _POSIX_PATH_MAX          255
126 #endif
127 #define _POSIX_PIPE_BUF          512
128 #define _POSIX_RTSIG_MAX         8
129 #define _POSIX_SEM_NSEMS_MAX     256
130 #define _POSIX_SEM_VALUE_MAX     32767
131 #define _POSIX_SIGQUEUE_MAX     32
132 #define _POSIX_SSIZE_MAX        32767
133 #define _POSIX_STREAM_MAX       8
134 #define _POSIX_TIMER_MAX        32
135 #ifdef _XPG6
136 #define _POSIX_TZNAME_MAX        6
137 #else
138 #define _POSIX_TZNAME_MAX        3 /* POSIX.1-1990 default */
139 #endif
140 /* POSIX.1c conformant */
141 #define _POSIX_LOGIN_NAME_MAX    9
142 #define _POSIX_THREAD_DESTRUCTOR_ITERATIONS 4
143 #define _POSIX_THREAD_KEYS_MAX  128
144 #define _POSIX_THREAD_THREADS_MAX 64
145 #define _POSIX_TTY_NAME_MAX     9
146 /* UNIX 03 conformant */
147 #define _POSIX_HOST_NAME_MAX    255
148 #define _POSIX_RE_DUP_MAX       255
149 #define _POSIX_SYMLINK_MAX      255
150 #define _POSIX_SYMLINK_MAX      8

152 /*
153 * POSIX.2 and XPG4-XSH4 conformant definitions
154 */

156 #define _POSIX2_BC_BASE_MAX      99
157 #define _POSIX2_BC_DIM_MAX      2048
158 #define _POSIX2_BC_SCALE_MAX    99
159 #define _POSIX2_BC_STRING_MAX   1000
160 #define _POSIX2_COLL_WEIGHTS_MAX 2
161 #define _POSIX2_EXPR_NEST_MAX   32
162 #define _POSIX2_LINE_MAX        2048
163 #define _POSIX2_RE_DUP_MAX      255
164 /* UNIX 03 conformant */
165 #define _POSIX2_CHARCLASS_NAME_MAX 14

167 #define BC_BASE_MAX              _POSIX2_BC_BASE_MAX
168 #define BC_DIM_MAX              _POSIX2_BC_DIM_MAX
169 #define BC_SCALE_MAX           _POSIX2_BC_SCALE_MAX
170 #define BC_STRING_MAX          _POSIX2_BC_STRING_MAX
171 #define COLL_WEIGHTS_MAX       10
172 #define EXPR_NEST_MAX          _POSIX2_EXPR_NEST_MAX
173 #define LINE_MAX               _POSIX2_LINE_MAX
174 #if !defined(_XPG6)
175 #define RE_DUP_MAX             _POSIX2_RE_DUP_MAX
176 #else
177 #define RE_DUP_MAX             _POSIX_RE_DUP_MAX
178 #endif /* !defined(_XPG6) */

180 #endif /* defined(__EXTENSIONS__) || !defined(_STRICT_STDC) ... */

182 #if defined(__EXTENSIONS__) || \
183     (!defined(_STRICT_STDC) && !defined(_POSIX_C_SOURCE)) || \
184     defined(_XOPEN_SOURCE)

186 /*
187 * For dual definitions for PASS_MAX and sysconf.c
188 */
189 #define _PASS_MAX_XPG         8 /* old standards PASS_MAX */
190 #define _PASS_MAX            256 /* modern Solaris PASS_MAX */

```

```

192 #if defined(_XPG3) && !defined(_XPG6)
193 #define PASS_MAX              _PASS_MAX_XPG /* max # of characters in a password */
194 #else /* XPG6 or just Solaris */
195 #define PASS_MAX              _PASS_MAX /* max # of characters in a password */
196 #endif /* defined(_XPG3) && !defined(_XPG6) */

198 #define CHARCLASS_NAME_MAX    _POSIX2_CHARCLASS_NAME_MAX

200 #define NL_ARGMAX             9 /* max value of "digit" in calls to the */
201 /* NLS printf() and scanf() */
202 #define NL_LANGMAX           14 /* max # of bytes in a LANG name */
203 #define NL_MSGMAX            32767 /* max message number */
204 #define NL_NMAX              1 /* max # bytes in N-to-1 mapping characters */
205 #define NL_SETMAX            255 /* max set number */
206 #define NL_TEXTMAX           2048 /* max set number */
207 #define NZERO                 20 /* default process priority */

209 #define WORD_BIT              32 /* # of bits in a "word" or "int" */
210 #if defined(_LP64)
211 #define LONG_BIT              64 /* # of bits in a "long" */
212 #else /* _ILP32 */
213 #define LONG_BIT              32 /* # of bits in a "long" */
214 #endif

216 /* Marked as LEGACY in SUSv2 and removed in UNIX 03 */
217 #ifndef _XPG6
218 #define DBL_DIG               15 /* digits of precision of a "double" */
219 #define DBL_MAX               1.7976931348623157081452E+308 /* max decimal value */
220 /* of a double */
221 #define FLT_DIG               6 /* digits of precision of a "float" */
222 #define FLT_MAX               3.4028234663852885981170E+38F /* max decimal value */
223 /* of a "float" */
224 #endif

226 /* Marked as LEGACY in SUSv1 and removed in SUSv2 */
227 #ifndef _XPG5
228 #define DBL_MIN               2.2250738585072013830903E-308 /* min decimal value */
229 /* of a double */
230 #define FLT_MIN               1.1754943508222875079688E-38F /* min decimal value */
231 /* of a float */
232 #endif

234 #endif /* defined(__EXTENSIONS__) || (!defined(_STRICT_STDC) ... */

236 #define _XOPEN_IOV_MAX        16 /* max # iovec/process with readv()/writev() */
237 #define _XOPEN_NAME_MAX      255 /* max # bytes in filename excluding null */
238 #define _XOPEN_PATH_MAX      1024 /* max # bytes in a pathname */

240 #define IOV_MAX               _XOPEN_IOV_MAX

242 #if defined(__EXTENSIONS__) || \
243     (!defined(_STRICT_STDC) && !defined(_XOPEN_OR_POSIX))

245 #define FCHR_MAX              1048576 /* max size of a file in bytes */
246 #define PID_MAX               999999 /* max value for a process ID */

248 /*
249 * POSIX 1003.1a, section 2.9.5, table 2-5 contains [NAME_MAX] and the
250 * related text states:
251 *
252 * A definition of one of the values from Table 2-5 shall be omitted from the
253 * <limits.h> on specific implementations where the corresponding value is
254 * equal to or greater than the stated minimum, but where the value can vary
255 * depending on the file to which it is applied. The actual value supported for
256 * a specific pathname shall be provided by the pathconf() (5.7.1) function.

```

```

257 *
258 * This is clear that any machine supporting multiple file system types
259 * and/or a network can not include this define, regardless of protection
260 * by the _POSIX_SOURCE and _POSIX_C_SOURCE flags.
261 *
262 * #define      NAME_MAX      14
263 */

265 #define CHILDD_MAX      25      /* max # of processes per user id */
266 #ifndef OPEN_MAX
267 #define OPEN_MAX      256      /* max # of files a process can have open */
268 #endif

270 #define PIPE_MAX      5120      /* max # bytes written to a pipe in a write */

272 #define STD_BLK      1024      /* # bytes in a physical I/O block */
273 #define UID_MAX      2147483647 /* max value for a user or group ID */
274 #define USI_MAX      4294967295 /* max decimal value of an "unsigned" */
275 #define SYSPID_MAX      1      /* max pid of system processes */

277 #ifndef SYS_NMLN
278 #define SYS_NMLN      257      /* also defined in sys/utsname.h */
279 #endif

281 #ifndef CLK_TCK

283 #if !defined(_CLOCK_T) || __cplusplus >= 199711L
284 #define _CLOCK_T
285 typedef long      clock_t;
286 #endif /* !_CLOCK_T */

288 extern long _sysconf(int); /* System Private interface to sysconf() */
289 #define CLK_TCK ((clock_t)_sysconf(3)) /* 3 is _SC_CLK_TCK */

291 #endif /* CLK_TCK */

293 #ifndef __USE_LEGACY_LOGNAME
294 #define LOGNAME_MAX      8      /* max # of characters in a login name */
295 #else /* __USE_LEGACY_LOGNAME */
296 #define LOGNAME_MAX      32      /* max # of characters in a login name */
297 /* Increased for illumos */
298 #endif /* __USE_LEGACY_LOGNAME */
299 #define LOGIN_NAME_MAX (LOGNAME_MAX + 1) /* max buffer size */
300 #define LOGNAME_MAX_TRAD      8 /* traditional length */
301 #define LOGIN_NAME_MAX_TRAD (LOGNAME_MAX_TRAD + 1) /* and size */

303 #define TTYNAME_MAX      128      /* max # of characters in a tty name */

305 #endif /* if defined(__EXTENSIONS__) || (!defined(_STRICT_STDC) ... */

307 #if defined(__EXTENSIONS__) || (_POSIX_C_SOURCE >= 199506L)
308 #include <sys/unistd.h>

310 #if !defined(_SIZE_T) || __cplusplus >= 199711L
311 #define _SIZE_T
312 #if defined(_LP64) || defined(_I32LPx)
313 typedef unsigned long size_t; /* size of something in bytes */
314 #else
315 typedef unsigned int size_t; /* (historical version) */
316 #endif
317 #endif /* _SIZE_T */

319 extern long _sysconf(int); /* System Private interface to sysconf() */

321 #define PTHREAD_STACK_MIN ((size_t)_sysconf(_SC_THREAD_STACK_MIN))
322 /* Added for UNIX98 conformance */

```

```

323 #define PTHREAD_DESTRUCTOR_ITERATIONS _POSIX_THREAD_DESTRUCTOR_ITERATIONS
324 #define PTHREAD_KEYS_MAX _POSIX_THREAD_KEYS_MAX
325 #define PTHREAD_THREADS_MAX _POSIX_THREAD_THREADS_MAX
326 #endif /* defined(__EXTENSIONS__) || (_POSIX_C_SOURCE >= 199506L) */

328 #ifdef __cplusplus
329 }

```

---

unchanged\_portion\_omitted

new/usr/src/head/stdlib.h

1

```
*****
11543 Sat Jun 1 10:02:50 2013
new/usr/src/head/stdlib.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
25 * Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
26 */

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

31 #ifndef _STDLIB_H
32 #define _STDLIB_H

34 #include <iso/stdlib_iso.h>
35 #include <iso/stdlib_c99.h>

37 #if defined(__EXTENSIONS__) || defined(_XPG4)
38 #include <sys/wait.h>
39 #endif

41 /*
42  * Allow global visibility for symbols defined in
43  * C++ "std" namespace in <iso/stdlib_iso.h>.
44  */
45 #if __cplusplus >= 199711L
46 using std::div_t;
47 using std::ldiv_t;
48 using std::size_t;
49 using std::abort;
50 using std::abs;
51 using std::atexit;
52 using std::atof;
53 using std::atoi;
54 using std::atol;
55 using std::bsearch;
56 using std::calloc;
57 using std::div;
58 using std::exit;
59 using std::free;
60 using std::getenv;
```

new/usr/src/head/stdlib.h

2

```
61 using std::labs;
62 using std::ldiv;
63 using std::malloc;
64 using std::mblen;
65 using std::mbstowcs;
66 using std::mbtowc;
67 using std::qsort;
68 using std::rand;
69 using std::realloc;
70 using std::srand;
71 using std::strtod;
72 using std::strtol;
73 using std::strtoul;
74 using std::system;
75 using std::wcstombs;
76 using std::wctomb;
77 #endif

79 #ifdef __cplusplus
80 extern "C" {
81 #endif

83 #ifndef _UID_T
84 #define _UID_T
85 typedef unsigned int    uid_t;          /* UID type          */
86 #endif /* !_UID_T */

88 #if defined(__STDC__)

90 /* large file compilation environment setup */
91 #if !defined(_LP64) && _FILE_OFFSET_BITS == 64

93 #ifdef __PRAGMA_REDEFINE_EXTNAME
94 #pragma redefine_extname    mkstemp        mkstemp64
95 #pragma redefine_extname    mkstemps      mkstemps64
96 #else /* __PRAGMA_REDEFINE_EXTNAME */
97 #define mkstemp              mkstemp64
98 #define mkstemps            mkstemps64
99 #endif /* __PRAGMA_REDEFINE_EXTNAME */

101 #endif /* _FILE_OFFSET_BITS == 64 */

103 /* In the LP64 compilation environment, all APIs are already large file */
104 #if defined(_LP64) && defined(_LARGEFILE64_SOURCE)

106 #ifdef __PRAGMA_REDEFINE_EXTNAME
107 #pragma redefine_extname    mkstemp64      mkstemp
108 #pragma redefine_extname    mkstemps64     mkstemps
109 #else /* __PRAGMA_REDEFINE_EXTNAME */
110 #define mkstemp64           mkstemp
111 #define mkstemps64         mkstemps
112 #endif /* __PRAGMA_REDEFINE_EXTNAME */

114 #endif /* _LP64 && _LARGEFILE64_SOURCE */

116 #if defined(__EXTENSIONS__) || \
117     (!defined(_STRICT_STDC) && !defined(_XOPEN_OR_POSIX)) || \
118     (_POSIX_C_SOURCE - 0 >= 199506L) || defined(_REENTRANT)
119 extern int rand_r(unsigned int *);
120 #endif

122 extern void _exithandle(void);

124 #if defined(__EXTENSIONS__) || \
125     (!defined(_STRICT_STDC) && !defined(_POSIX_C_SOURCE)) || \
126     defined(_XPG4)
```

```

127 extern double drand48(void);
128 extern double erand48(unsigned short *);
129 extern long jrand48(unsigned short *);
130 extern void lcong48(unsigned short *);
131 extern long lrand48(void);
132 extern long mrand48(void);
133 extern long nrand48(unsigned short *);
134 extern unsigned short *seed48(unsigned short *);
135 extern void srand48(long);
136 extern int putenv(char *);
137 extern void setkey(const char *);
138 #endif /* defined(__EXTENSIONS__) || !defined(_STRICT_STDC) ... */

140 /*
141 * swab() has historically been in <stdlib.h> as delivered from AT&T
142 * and continues to be visible in the default compilation environment.
143 * As of Issue 4 of the X/Open Portability Guides, swab() was declared
144 * in <unistd.h>. As a result, with respect to X/Open namespace the
145 * swab() declaration in this header is only visible for the XPG3
146 * environment.
147 */
148 #if defined(__EXTENSIONS__) || \
149     (!defined(_STRICT_STDC) && !defined(_POSIX_C_SOURCE)) && \
150     (!defined(_XOPEN_SOURCE) || (defined(XPG3) && !defined(XPG4)))
151 #ifndef _SSIZE_T
152 #define _SSIZE_T
153 #if defined(_LP64) || defined(_I32LPx)
154 typedef long    ssize_t;    /* size of something in bytes or -1 */
155 #else
156 typedef int     ssize_t;    /* (historical version) */
157 #endif
158 #endif /* !_SSIZE_T */

160 extern void swab(const char *, char *, ssize_t);
161 #endif /* defined(__EXTENSIONS__) || !defined(_STRICT_STDC) ... */

163 #if defined(__EXTENSIONS__) || \
164     (!defined(_XOPEN_OR_POSIX) || defined(XPG4_2) || \
165     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64))
166 extern int    mkstemp(char *);
167 #if !defined(XPG4_2) || defined(__EXTENSIONS__)
168 extern int    mkstemps(char *, int);
169 #endif
170 #endif /* defined(__EXTENSIONS__) ... */

172 #if defined(_LARGEFILE64_SOURCE) && !((_FILE_OFFSET_BITS == 64) && \
173     !defined(__PRAGMA_REDEFINE_EXTNAME))
174 extern int    mkstemp64(char *);
175 #if !defined(XPG4_2) || defined(__EXTENSIONS__)
176 extern int    mkstemps64(char *, int);
177 #endif
178 #endif /* _LARGEFILE64_SOURCE... */

180 #if defined(__EXTENSIONS__) || \
181     (!defined(_STRICT_STDC) && !defined(_XOPEN_OR_POSIX)) || \
182     defined(XPG4_2)
183 extern long a64l(const char *);
184 extern char *ecvt(double, int, int *RESTRICT_KYWD, int *RESTRICT_KYWD);
185 extern char *fcvt(double, int, int *RESTRICT_KYWD, int *RESTRICT_KYWD);
186 extern char *gcvt(double, int, char *);
187 extern int getsuopt(char **, char *const *, char **);
188 extern int grantpt(int);
189 extern char *initstate(unsigned, char *, size_t);
190 extern char *l64a(long);
191 extern char *mktemp(char *);
192 extern char *ptsname(int);

```

```

193 extern long random(void);
194 extern char *realpath(const char *_RESTRICT_KYWD, char *_RESTRICT_KYWD);
195 extern char *setstate(const char *);
196 extern void srandom(unsigned);
197 extern int unlockpt(int);
198 /* Marked LEGACY in SUSv2 and removed in SUSv3 */
199 #if !defined(XPG6) || defined(__EXTENSIONS__)
200 extern int ttyslot(void);
201 extern void *valloc(size_t);
202 #endif /* !defined(XPG6) || defined(__EXTENSIONS__) */
203 #endif /* defined(__EXTENSIONS__) || ... || defined(XPG4_2) */

205 #if defined(__EXTENSIONS__) || \
206     (!defined(_STRICT_STDC) && !defined(_XOPEN_OR_POSIX)) || \
207     defined(XPG6)
208 extern int posix_memalign(void **, size_t, size_t);
209 extern int posix_openpt(int);
210 extern int setenv(const char *, const char *, int);
211 extern int unsetenv(const char *);
212 #endif

214 #if defined(__EXTENSIONS__) || \
215     (!defined(_STRICT_STDC) && !defined(_XOPEN_OR_POSIX))
216 extern char *canonicalize_file_name(const char *);
217 extern int clearenv(void);
218 extern void closefrom(int);
219 extern int daemon(int, int);
220 extern int dup2(int, int);
221 extern int fdwalk(int (*)(void *, int), void *);
222 extern char *qecvt(long double, int, int *, int *);
223 extern char *qfcvt(long double, int, int *, int *);
224 extern char *qgcvt(long double, int, char *);
225 extern char *getcwd(char *, size_t);
226 extern const char *getexecname(void);

228 #ifndef __GETLOGIN_DEFINED /* Avoid duplicate in unistd.h */
229 #define __GETLOGIN_DEFINED
230 #ifndef USE_LEGACY_LOGNAME
231 #ifdef __PRAGMA_REDEFINE_EXTNAME
232 #pragma redefine_extname getlogin getloginx
233 #else /* __PRAGMA_REDEFINE_EXTNAME */
234 extern char *getloginx(void);
235 #define getlogin    getloginx
236 #endif /* __PRAGMA_REDEFINE_EXTNAME */
237 #endif /* USE_LEGACY_LOGNAME */
238 extern char *getlogin(void);
239 #endif /* __GETLOGIN_DEFINED */

241 extern int getopt(int, char *const *, const char *);
242 extern char *optarg;
243 extern int optind, opterr, optopt;
244 extern char *getpass(const char *);
245 extern char *getpassphrase(const char *);
246 extern int getpw(uid_t, char *);
247 extern int isatty(int);
248 extern void *memalign(size_t, size_t);
249 extern char *ttyname(int);
250 extern char *mkdtemp(char *);
251 extern const char *getprogname(void);
252 extern void setprogname(const char *);

254 #if !defined(_STRICT_STDC) && defined(_LONGLONG_TYPE)
255 extern char *lltostr(long long, char *);
256 extern char *ulltostr(unsigned long long, char *);
257 #endif /* !defined(_STRICT_STDC) && defined(_LONGLONG_TYPE) */

```



```

259 #endif /* defined(__EXTENSIONS__) || !defined(__STRICT_STDC) ... */
261 #else /* not __STDC__ */
263 #if defined(__EXTENSIONS__) || !defined(__XOPEN_OR_POSIX) || \
264     (_POSIX_C_SOURCE - 0 >= 199506L) || defined(_REENTRANT)
265 extern int rand_r();
266 #endif /* defined(__EXTENSIONS__) || defined(_REENTRANT) ... */
268 extern void _exithandle();
270 #if defined(__EXTENSIONS__) || !defined(_POSIX_C_SOURCE) || defined(_XPG4)
271 extern double drand48();
272 extern double erand48();
273 extern long jrand48();
274 extern void lcong48();
275 extern long lrand48();
276 extern long mrand48();
277 extern long nrand48();
278 extern unsigned short *seed48();
279 extern void srand48();
280 extern int putenv();
281 extern void setkey();
282 #endif /* defined(__EXTENSIONS__) || !defined(_POSIX_C_SOURCE) ... */
284 #if (defined(__EXTENSIONS__) || !defined(_POSIX_C_SOURCE)) && \
285     (!defined(__XOPEN_SOURCE) || (defined(_XPG3) && !defined(_XPG4)))
286 extern void swab();
287 #endif
289 #if defined(__EXTENSIONS__) || \
290     !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || \
291     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64)
292 extern int mkstemp();
293 #if !defined(_XPG4_2) || defined(__EXTENSIONS__)
294 extern int mkstemps();
295 #endif
296 #endif /* defined(__EXTENSIONS__) ... */
298 #if defined(_LARGEFILE64_SOURCE) && !((_FILE_OFFSET_BITS == 64) && \
299     !defined(__PRAGMA_REDEFINE_EXTNAME))
300 extern int mkstemp64();
301 #if !defined(_XPG4_2) || defined(__EXTENSIONS__)
302 extern int mkstemps64();
303 #endif
304 #endif /* _LARGEFILE64_SOURCE... */
306 #if defined(__EXTENSIONS__) || !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)
307 extern long a64l();
308 extern char *ecvt();
309 extern char *fcvt();
310 extern char *gcvt();
311 extern int getsuopt();
312 extern int grantpt();
313 extern char *initstate();
314 extern char *l64a();
315 extern char *mktemp();
316 extern char *ptsname();
317 extern long random();
318 extern char *realpath();
319 extern char *setstate();
320 extern void srandom();
321 /* Marked LEGACY in SUSv2 and removed in SUSv3 */
322 #if !defined(_XPG6) || defined(__EXTENSIONS__)
323 extern int ttyslot();
324 extern void *valloc();

```

```

325 #endif /* !defined(_XPG6) || defined(__EXTENSIONS__) */
326 #endif /* defined(__EXTENSIONS__) || ... || defined(_XPG4_2) */
328 #if defined(__EXTENSIONS__) || !defined(__XOPEN_OR_POSIX) || defined(_XPG6)
329 extern int posix_memalign();
330 extern int posix_openpt();
331 extern int setenv();
332 extern int unsetenv();
333 #endif
335 #if defined(__EXTENSIONS__) || !defined(__XOPEN_OR_POSIX)
336 extern char *canonicalize_file_name();
337 extern int clearenv();
338 extern void closefrom();
339 extern int daemon();
340 extern int dup2();
341 extern int fdwalk();
342 extern char *qecvt();
343 extern char *qfcvt();
344 extern char *ggcvt();
345 extern char *getcwd();
346 extern char *getexecname();
348 #ifndef __GETLOGIN_DEFINED /* Avoid duplicate in unistd.h */
349 #define __GETLOGIN_DEFINED
350 #ifndef USE_LEGACY_LOGNAME
351 #ifdef __PRAGMA_REDEFINE_EXTNAME
352 #pragma redefine_extname getlogin getloginx
353 #else /* __PRAGMA_REDEFINE_EXTNAME */
354 extern char *getloginx();
355 #define getlogin getloginx
356 #endif /* __PRAGMA_REDEFINE_EXTNAME */
357 #endif /* USE_LEGACY_LOGNAME */
358 extern char *getlogin();
359 #endif /* __GETLOGIN_DEFINED */
361 extern int getopt();
362 extern char *optarg;
363 extern int optind, opterr, optopt;
364 extern char *getpass();
365 extern char *getpassphrase();
366 extern int getpw();
367 extern int isatty();
368 extern void *memalign();
369 extern char *ttyname();
370 extern char *mkdtemp();
371 extern char *getprogname();
372 extern void setprogname();
374 #if defined(_LONGLONG_TYPE)
375 extern char *lltostr();
376 extern char *ulltostr();
377 #endif /* defined(_LONGLONG_TYPE) */
378 #endif /* defined(__EXTENSIONS__) || !defined(__XOPEN_OR_POSIX) ... */
380 #endif /* __STDC__ */
382 #ifdef __cplusplus
383 }
    unchanged_portion_omitted

```

new/usr/src/head/unistd.h

1

```
*****
40480 Sat Jun 1 10:02:50 2013
new/usr/src/head/unistd.h
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 *
25 * Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
26 */

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

31 #ifndef _UNISTD_H
32 #define _UNISTD_H

34 #include <sys/feature_tests.h>

36 #include <sys/types.h>
37 #include <sys/unistd.h>

39 #ifdef __cplusplus
40 extern "C" {
41 #endif

43 /* Symbolic constants for the "access" routine: */
44 #define R_OK 4 /* Test for Read permission */
45 #define W_OK 2 /* Test for Write permission */
46 #define X_OK 1 /* Test for eXecute permission */
47 #define F_OK 0 /* Test for existence of File */

49 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
50 #define F_ULOCK 0 /* Unlock a previously locked region */
51 #define F_LOCK 1 /* Lock a region for exclusive use */
52 #define F_TLOCK 2 /* Test and lock a region for exclusive use */
53 #define F_TEST 3 /* Test a region for other processes locks */
54 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */

56 /* Symbolic constants for the "lseek" routine: */

58 #ifndef SEEK_SET
59 #define SEEK_SET 0 /* Set file pointer to "offset" */
60 #endif
```

new/usr/src/head/unistd.h

2

```
62 #ifndef SEEK_CUR
63 #define SEEK_CUR 1 /* Set file pointer to current plus "offset" */
64 #endif

66 #ifndef SEEK_END
67 #define SEEK_END 2 /* Set file pointer to EOF plus "offset" */
68 #endif

70 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
71 #ifndef SEEK_DATA
72 #define SEEK_DATA 3 /* Set file pointer to next data past offset */
73 #endif

75 #ifndef SEEK_HOLE
76 #define SEEK_HOLE 4 /* Set file pointer to next hole past offset */
77 #endif
78 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

80 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
81 /* Path names: */
82 #define GF_PATH "/etc/group" /* Path name of the "group" file */
83 #define PF_PATH "/etc/passwd" /* Path name of the "passwd" file */
84 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

86 /*
87 * compile-time symbolic constants,
88 * Support does not mean the feature is enabled.
89 * Use pathconf/sysconf to obtain actual configuration value.
90 */

92 /* Values unchanged in UNIX 03 */
93 #define _POSIX_ASYNC_IO 1
94 #define _POSIX_JOB_CONTROL 1
95 #define _POSIX_SAVED_IDS 1
96 #define _POSIX_SYNC_IO 1

98 /*
99 * POSIX.1b compile-time symbolic constants.
100 */
101 #if defined(_XPG6)
102 #define _POSIX_ASYNCHRONOUS_IO 200112L
103 #define _POSIX_FSYNC 200112L
104 #define _POSIX_MAPPED_FILES 200112L
105 #define _POSIX_MEMLOCK 200112L
106 #define _POSIX_MEMLOCK_RANGE 200112L
107 #define _POSIX_MEMORY_PROTECTION 200112L
108 #define _POSIX_MESSAGE_PASSING 200112L
109 #define _POSIX_PRIORITY_SCHEDULING 200112L
110 #define _POSIX_REALTIME_SIGNALS 200112L
111 #define _POSIX_SEMAPHORES 200112L
112 #define _POSIX_SHARED_MEMORY_OBJECTS 200112L
113 #define _POSIX_SYNCHRONIZED_IO 200112L
114 #else
115 #define _POSIX_ASYNCHRONOUS_IO 1
116 #define _POSIX_FSYNC 1
117 #define _POSIX_MAPPED_FILES 1
118 #define _POSIX_MEMLOCK 1
119 #define _POSIX_MEMLOCK_RANGE 1
120 #define _POSIX_MEMORY_PROTECTION 1
121 #define _POSIX_MESSAGE_PASSING 1
122 #define _POSIX_PRIORITY_SCHEDULING 1
123 #define _POSIX_REALTIME_SIGNALS 1
124 #define _POSIX_SEMAPHORES 1
125 #define _POSIX_SHARED_MEMORY_OBJECTS 1
126 #define _POSIX_SYNCHRONIZED_IO 1
```

```

127 #endif

129 /*
130 * POSIX.1c compile-time symbolic constants.
131 */
132 #if defined(_XPG6)
133 #define _POSIX_THREAD_SAFE_FUNCTIONS      200112L
134 #define _POSIX_THREADS                    200112L
135 #define _POSIX_THREAD_ATTR_STACKADDR     200112L
136 #define _POSIX_THREAD_ATTR_STACKSIZE    200112L
137 #define _POSIX_THREAD_PROCESS_SHARED     200112L
138 #define _POSIX_THREAD_PRIORITY_SCHEDULING 200112L
139 #define _POSIX_TIMERS                    200112L
140 #else
141 #define _POSIX_THREAD_SAFE_FUNCTIONS      1
142 #define _POSIX_THREADS                    1
143 #define _POSIX_THREAD_ATTR_STACKADDR     1
144 #define _POSIX_THREAD_ATTR_STACKSIZE    1
145 #define _POSIX_THREAD_PROCESS_SHARED     1
146 #define _POSIX_THREAD_PRIORITY_SCHEDULING 1
147 #define _POSIX_TIMERS                    1
148 #endif

150 /* New in UNIX 03 */
151 #define _POSIX_ADVISORY_INFO              200112L
152 #define _POSIX_BARRIERS                  200112L
153 #define _POSIX_CLOCK_SELECTION           200112L
154 #define _POSIX_IPV6                      200112L
155 #define _POSIX_MONOTONIC_CLOCK           200112L
156 #define _POSIX_RAW_SOCKETS               200112L
157 #define _POSIX_READER_WRITER_LOCKS      200112L
158 #define _POSIX_SPAWN                     200112L
159 #define _POSIX_SPIN_LOCKS                200112L
160 #define _POSIX_TIMEOUTS                  200112L

162 /*
163 * Support for the POSIX.1 mutex protocol attribute. For realtime applications
164 * which need mutexes to support priority inheritance/ceiling.
165 */
166 #if defined(_XPG6)
167 #define _POSIX_THREAD_PRIO_INHERIT        200112L
168 #define _POSIX_THREAD_PRIO_PROTECT       200112L
169 #else
170 #define _POSIX_THREAD_PRIO_INHERIT        1
171 #define _POSIX_THREAD_PRIO_PROTECT       1
172 #endif

174 #ifndef _POSIX_VDISABLE
175 #define _POSIX_VDISABLE                   0
176 #endif

178 #ifndef NULL
179 #if defined(_LP64)
180 #define NULL      0L
181 #else
182 #define NULL      0
183 #endif
184 #endif

186 #define STDIN_FILENO    0
187 #define STDOUT_FILENO  1
188 #define STDERR_FILENO  2

190 /*
191 * Large File Summit-related announcement macros. The system supports both
192 * the additional and transitional Large File Summit interfaces. (The final

```

```

193 * two macros provide a finer granularity breakdown of _LFS64_LARGEFILE.)
194 */
195 #define _LFS_LARGEFILE      1
196 #define _LFS64_LARGEFILE   1
197 #define _LFS64_STDIO       1
198 #define _LFS64_ASYNCHRONOUS_IO 1

200 /* large file compilation environment setup */
201 #if !defined(_LP64) && _FILE_OFFSET_BITS == 64
202 #ifdef __PRAGMA_REDEFINE_EXTNAME
203 #pragma redefine_extname      ftruncate      ftruncate64
204 #pragma redefine_extname      lseek          lseek64
205 #pragma redefine_extname      pread          pread64
206 #pragma redefine_extname      pwrite        pwrite64
207 #pragma redefine_extname      truncate      truncate64
208 #pragma redefine_extname      lockf        lockf64
209 #pragma redefine_extname      tell         tell64
210 #else /* __PRAGMA_REDEFINE_EXTNAME */
211 #define ftruncate              ftruncate64
212 #define lseek                  lseek64
213 #define pread                  pread64
214 #define pwrite                  pwrite64
215 #define truncate              truncate64
216 #define lockf                  lockf64
217 #define tell                    tell64
218 #endif /* __PRAGMA_REDEFINE_EXTNAME */
219 #endif /* !_LP64 && _FILE_OFFSET_BITS == 64 */

221 /* In the LP64 compilation environment, the APIs are already large file */
222 #if defined(_LP64) && defined(_LARGEFILE64_SOURCE)
223 #ifdef __PRAGMA_REDEFINE_EXTNAME
224 #pragma redefine_extname      ftruncate64      ftruncate
225 #pragma redefine_extname      lseek64          lseek
226 #pragma redefine_extname      pread64          pread
227 #pragma redefine_extname      pwrite64         pwrite
228 #pragma redefine_extname      truncate64       truncate
229 #pragma redefine_extname      lockf64         lockf
230 #pragma redefine_extname      tell64         tell
231 #else /* __PRAGMA_REDEFINE_EXTNAME */
232 #define ftruncate64            ftruncate
233 #define lseek64                lseek
234 #define pread64                pread
235 #define pwrite64                pwrite
236 #define truncate64            truncate
237 #define lockf64                lockf
238 #define tell64                 tell
239 #endif /* __PRAGMA_REDEFINE_EXTNAME */
240 #endif /* !_LP64 && _LARGEFILE64_SOURCE */

242 #if defined(__STDC__)

244 extern int access(const char *, int);
245 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
246 extern int acct(const char *);
247 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
248 extern unsigned alarm(unsigned);
249 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
250 #if !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
251     defined(__EXTENSIONS__)
252 extern int brk(void *);
253 #endif /* !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2)... */
254 extern int chdir(const char *);
255 extern int chown(const char *, uid_t, gid_t);
256 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
257 #if !defined(_POSIX_C_SOURCE) || (defined(_XOPEN_SOURCE) && \
258     !defined(_XPG6)) || defined(__EXTENSIONS__)

```

```

259 extern int chroot(const char *);
260 #endif /* !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE)... */
261 extern int close(int);
262 #if defined(_XPG4) || defined(__EXTENSIONS__)
263 extern size_t confstr(int, char *, size_t);
264 extern char *crypt(const char *, const char *);
265 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
266 #if !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE) || \
267     defined(__EXTENSIONS__)
268 extern char *ctermid(char *);
269 #endif /* (!defined(_POSIX_C_SOURCE) ... */
270 #if !defined(__XOPEN_OR_POSIX) || defined(_REENTRANT) || defined(__EXTENSIONS__)
271 extern char *ctermid_r(char *);
272 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_REENTRANT) ... */
273 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
274 #if !defined(_XPG6) || defined(__EXTENSIONS__)
275 extern char *cuserid(char *);
276 #endif
277 extern int dup(int);
278 extern int dup2(int, int);
279 #if defined(_XPG4) || defined(__EXTENSIONS__)
280 extern void encrypt(char *, int);
281 #endif /* defined(XPG4) || defined(__EXTENSIONS__) */
282 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
283 extern void endusershell(void);
284 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
285 extern int execl(const char *, const char *, ...);
286 extern int execlp(const char *, const char *, ...);
287 extern int execlp(const char *, const char *, ...);
288 extern int execv(const char *, char *const *);
289 extern int execvp(const char *, char *const *, char *const *);
290 extern int execvp(const char *, char *const *);
291 extern void _exit(int);
292     _NORETURN;
293 /*
294  * The following fattach prototype is duplicated in <stropts.h>. The
295  * duplication is necessitated by XPG4.2 which requires the prototype
296  * be defined in <stropts.h>.
297  */
298 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
299 extern int fattach(int, const char *);
300 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
301 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
302 extern int fchdir(int);
303 extern int fchown(int, uid_t, gid_t);
304 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
305 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
306 extern int fchroot(int);
307 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
308 #if !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || \
309     defined(__EXTENSIONS__)
310 extern int fdatsync(int);
311 #endif /* !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2)... */
312 /*
313  * The following fdetach prototype is duplicated in <stropts.h>. The
314  * duplication is necessitated by XPG4.2 which requires the prototype
315  * be defined in <stropts.h>.
316  */
317 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
318 extern int fdetach(const char *);
319 #endif /* !defined(__XOPEN_OR_POSIX)... */
320 extern pid_t fork(void);
321 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
322 extern pid_t fork1(void);
323 extern pid_t forkall(void);
324 #endif /* !defined(__XOPEN_OR_POSIX)... */

```

```

325 extern long fpathconf(int, int);
326 #if !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2) || \
327     defined(__EXTENSIONS__)
328 extern int fsync(int);
329 #endif /* !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2)... */
330 #if !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || defined(_XPG4_2) || \
331     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
332     defined(__EXTENSIONS__)
333 extern int ftruncate(int, off_t);
334 #endif /* !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2)... */
335 extern char *getcwd(char *, size_t);
336 #if !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
337     defined(__EXTENSIONS__)
338 extern int getdtablesize(void);
339 #endif
340 extern gid_t getegid(void);
341 extern uid_t geteuid(void);
342 extern gid_t getgid(void);
343 extern int getgroups(int, gid_t *);
344 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
345 extern long gethostid(void);
346 #endif
347 #if defined(_XPG4_2)
348 extern int gethostname(char *, size_t);
349 #elif !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
350 extern int gethostname(char *, int);
351 #endif

353 #ifndef __GETLOGIN_DEFINED /* Avoid duplicate in stdlib.h */
354 #define __GETLOGIN_DEFINED
355 #ifndef USE_LEGACY_LOGNAME
356 #ifdef __PRAGMA_REDEFINE_EXTNAME
357 #pragma redefine_extname getlogin getloginx
358 #else /* __PRAGMA_REDEFINE_EXTNAME */
359 extern char *getlogin(void);
360 #define getlogin getloginx
361 #endif /* __PRAGMA_REDEFINE_EXTNAME */
362 #endif /* USE_LEGACY_LOGNAME */
363 extern char *getlogin(void);
364 #endif /* __GETLOGIN_DEFINED */

366 #if defined(_XPG4) || defined(__EXTENSIONS__)
367 extern int getopt(int, char *const *, const char *);
368 extern char *optarg;
369 extern int opterr, optind, optopt;
370 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
371 #if !defined(_XPG6) || defined(__EXTENSIONS__)
372 extern char *getpass(const char *);
373 #endif
374 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
375 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
376 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
377 #if !defined(_XPG6) || defined(__EXTENSIONS__)
378 extern int getpagesize(void);
379 #endif
380 extern pid_t getpgid(pid_t);
381 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
382 extern pid_t getpid(void);
383 extern pid_t getppid(void);
384 extern pid_t getpgrp(void);

386 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
387 char *gettext(const char *, const char *);
388 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
389 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
390 extern pid_t getsid(pid_t);

```

```

391 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
392 extern uid_t getuid(void);
393 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
394 extern char *getusershell(void);
395 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
396 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
397 extern char *getwd(char *);
398 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
399 /*
400  * The following ioctl prototype is duplicated in <stropts.h>. The
401  * duplication is necessitated by XPG4.2 which requires the prototype
402  * be defined in <stropts.h>.
403  */
404 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
405 extern int ioctl(int, int, ...);
406 extern int isaexec(const char *, char *const *, char *const *);
407 extern int issetugid(void);
408 #endif
409 extern int isatty(int);
410 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
411 extern int lchown(const char *, uid_t, gid_t);
412 #endif
413 extern int link(const char *, const char *);
414 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
415 extern off_t llseek(int, off_t, int);
416 #endif
417 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || \
418     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
419     defined(__EXTENSIONS__)
420 extern int lockf(int, int, off_t);
421 #endif
422 extern off_t lseek(int, off_t, int);
423 #if !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE) || \
424     defined(__EXTENSIONS__)
425 extern int nice(int);
426 #endif /* !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE)... */
427 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
428 extern int mincore(caddr_t, size_t, char *);
429 #endif
430 extern long pathconf(const char *, int);
431 extern int pause(void);
432 extern int pipe(int *);
433 #if !defined(_POSIX_C_SOURCE) || defined(_XPG5) || \
434     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
435     defined(__EXTENSIONS__)
436 extern ssize_t pread(int, void *, size_t, off_t);
437 #endif
438 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
439 extern void profil(unsigned short *, size_t, unsigned long, unsigned int);
440 #endif
441 /*
442  * pthread_atfork() is also declared in <pthread.h> as per SUSv3. The
443  * declarations are identical. A change to either one may also require
444  * appropriate namespace updates in order to avoid redeclaration
445  * warnings in the case where both prototypes are exposed via inclusion
446  * of both <pthread.h> and <unistd.h>.
447  */
448 #if !defined(__XOPEN_OR_POSIX) || \
449     ((_POSIX_C_SOURCE > 2) && !defined(_XPG6)) || \
450     defined(__EXTENSIONS__)
451 extern int pthread_atfork(void (*) (void), void (*) (void), void (*) (void));
452 #endif /* !defined(__XOPEN_OR_POSIX) || ((_POSIX_C_SOURCE > 2) ... */
453 #if !defined(_LP64) && \
454     (!defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__))
455 extern int ptrace(int, pid_t, int, int);
456 #endif

```

```

457 #if !defined(_POSIX_C_SOURCE) || defined(_XPG5) || \
458     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
459     defined(__EXTENSIONS__)
460 extern ssize_t pwrite(int, const void *, size_t, off_t);
461 #endif
462 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
463 /* per RFC 3542; This is also defined in netdb.h */
464 extern int rcmd_af(char **, unsigned short, const char *, const char *,
465     const char *, int *, int);
466 #endif
467 extern ssize_t read(int, void *, size_t);
468 #if !defined(__XOPEN_OR_POSIX) || \
469     defined(_XPG4_2) || defined(__EXTENSIONS__)
470 extern ssize_t readlink(const char *_RESTRICT_KYWD, char *_RESTRICT_KYWD,
471     size_t);
472 #endif
473 #if (!defined(__XOPEN_OR_POSIX) || (defined(_XPG3) && !defined(_XPG4))) || \
474     defined(__EXTENSIONS__)
475 extern int rename(const char *, const char *);
476 #endif /* (!defined(__XOPEN_OR_POSIX) || (defined(_XPG3)... */
477 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
478 extern int resolvepath(const char *, char *, size_t);
479 /* per RFC 3542; This is also defined in netdb.h */
480 extern int rexec_af(char **, unsigned short, const char *, const char *,
481     const char *, int *, int);
482 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
483 extern int rmdir(const char *);
484 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
485 /* per RFC 3542; This is also defined in netdb.h */
486 extern int rresvport_af(int *, int);
487 #endif
488
489 #if !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
490     defined(__EXTENSIONS__)
491 extern void *sbrk(intptr_t);
492 #endif /* !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2)... */
493 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG6) || defined(__EXTENSIONS__)
494 extern int setgid(gid_t);
495 extern int seteuid(uid_t);
496 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG6) ... */
497 extern int setgid(gid_t);
498 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
499 extern int setgroups(int, const gid_t *);
500 extern int sethostname(char *, int);
501 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
502 extern int setpgid(pid_t, pid_t);
503 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
504 extern pid_t setpgrp(void);
505 extern int setregid(gid_t, gid_t);
506 extern int setreuid(uid_t, uid_t);
507 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
508 extern pid_t setsid(void);
509 extern int setuid(uid_t);
510 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
511 extern void setusershell(void);
512 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
513 extern unsigned sleep(unsigned);
514 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
515 extern int stime(const time_t *);
516 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
517 #if defined(_XPG4)
518 /* __EXTENSIONS__ makes the SVID Third Edition prototype in stdlib.h visible */
519 extern void swab(const void *_RESTRICT_KYWD, void *_RESTRICT_KYWD, ssize_t);
520 #endif /* defined(_XPG4) */
521 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
522 extern int symlink(const char *, const char *);

```

```

523 extern void sync(void);
524 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) */
525 #if defined(_XPG5) && !defined(_XPG6)
526 #ifdef __PRAGMA_REDEFINE_EXTNAME
527 #pragma redefine_extname sysconf __sysconf_xpg5
528 #else /* __PRAGMA_REDEFINE_EXTNAME */
529 #define sysconf __sysconf_xpg5
530 #endif /* __PRAGMA_REDEFINE_EXTNAME */
531 #endif /* defined(_XPG5) && !defined(_XPG6) */
532 extern long sysconf(int);
533 extern pid_t tcgetpgrp(int);
534 extern int tcsetpgrp(int, pid_t);
535 #if !defined(__XOPEN_OR_POSIX) || \
536     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
537     defined(__EXTENSIONS__)
538 extern off_t tell(int);
539 #endif /* !defined(__XOPEN_OR_POSIX)... */
540 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || \
541     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
542     defined(__EXTENSIONS__)
543 extern int truncate(const char *, off_t);
544 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
545 extern char *ttyname(int);
546 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
547 extern useconds_t ualarm(useconds_t, useconds_t);
548 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
549 extern int unlink(const char *);
550 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
551 extern int usleep(useconds_t);
552 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
553 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
554 extern pid_t vfork(void) __RETURNS_TWICE;
555 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
556 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
557 extern void vhangup(void);
558 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
559 extern ssize_t write(int, const void *, size_t);
560 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
561 extern void yield(void);
562 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

564 #if !defined(__XOPEN_OR_POSIX) || defined(_ATFILE_SOURCE) || \
565     defined(__EXTENSIONS__)
566     /* || defined(_XPG7) */
567 extern int faccessat(int, const char *, int, int);
568 extern int fchownat(int, const char *, uid_t, gid_t, int);
569 extern int linkat(int, const char *, int, const char *, int);
570 extern ssize_t readlinkat(int, const char *_RESTRICT_KYWD,
571     char *_RESTRICT_KYWD, size_t);
572 extern int renameat(int, const char *, int, const char *);
573 extern int symlinkat(const char *, int, const char *);
574 extern int unlinkat(int, const char *, int);
575 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_ATFILE_SOURCE)... */
576 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
577 extern int get_nprocs(void);
578 extern int get_nprocs_conf(void);
579 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

581 /* transitional large file interface versions */
582 #if defined(_LARGEFILE64_SOURCE) && !((_FILE_OFFSET_BITS == 64) && \
583     !defined(__PRAGMA_REDEFINE_EXTNAME))
584 extern int ftruncate64(int, off64_t);
585 extern off64_t lseek64(int, off64_t, int);
586 extern ssize_t pread64(int, void *, size_t, off64_t);
587 extern ssize_t pwrite64(int, const void *, size_t, off64_t);
588 extern off64_t tell64(int);

```

```

589 extern int truncate64(const char *, off64_t);
590 extern int lockf64(int, int, off64_t);
591 #endif /* _LARGEFILE64_SOURCE */

593 #else /* __STDC__ */

595 extern int access();
596 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
597 extern int acct();
598 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
599 extern unsigned alarm();
600 #if !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
601     defined(__EXTENSIONS__)
602 extern int brk();
603 #endif /* !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2)... */
604 extern int chdir();
605 extern int chown();
606 #if !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE) || \
607     defined(__EXTENSIONS__)
608 extern int chroot();
609 #endif /* !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE)... */
610 extern int close();
611 #if defined(_XPG4) || defined(__EXTENSIONS__)
612 extern size_t confstr();
613 extern char *crypt();
614 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
615 #if !defined(_POSIX_C_SOURCE) || defined(_XPG3) || defined(__EXTENSIONS__)
616 extern char *ctermid();
617 #endif /* (!defined(_POSIX_C_SOURCE) || defined(_XPG3))... */
618 #if !defined(__XOPEN_OR_POSIX) || defined(_REENTRANT) || defined(__EXTENSIONS__)
619 extern char *ctermid_r();
620 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_REENTRANT) ... */
621 #if !defined(_XPG6) || defined(__EXTENSIONS__)
622 extern char *cuserid();
623 #endif
624 extern int dup();
625 extern int dup2();
626 #if defined(_XPG4) || defined(__EXTENSIONS__)
627 extern void encrypt();
628 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
629 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
630 extern void endusershell();
631 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
632 extern int execl();
633 extern int execlp();
634 extern int execlp();
635 extern int execv();
636 extern int execve();
637 extern int execvp();
638 extern void _exit();
639 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
640 extern int fattach();
641 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
642 #if !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
643 extern int fchdir();
644 extern int fchown();
645 #endif /* !defined(__XOPEN_OR_POSIX) || defined(_XPG4_2)... */
646 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
647 extern int fchroot();
648 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
649 #if !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || \
650     defined(__EXTENSIONS__)
651 extern int fdatsync();
652 #endif /* !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2)... */
653 #if !defined(__XOPEN_OR_POSIX)
654 extern int fdetach();

```

```

655 #endif /* !defined(__XOPEN_OR_POSIX) */
656 extern pid_t fork();
657 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
658 extern pid_t fork1();
659 extern pid_t forkall();
660 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
661 extern long fpathconf();
662 #if !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2) || \
663     defined(__EXTENSIONS__)
664 extern int fsync();
665 #endif /* !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2)... */
666 #if !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || defined(XPG4_2) || \
667     (defined(LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
668     defined(__EXTENSIONS__)
669 extern int ftruncate();
670 #endif /* !defined(__XOPEN_OR_POSIX) (_POSIX_C_SOURCE > 2)... */
671 extern char *getcwd();
672 #if !defined(__XOPEN_OR_POSIX) || (defined(XPG4_2) && !defined(XPG6)) || \
673     defined(__EXTENSIONS__)
674 extern int getdtablesize();
675 #endif
676 extern gid_t getegid();
677 extern uid_t geteuid();
678 extern gid_t getgid();
679 extern int getgroups();
680 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)
681 extern long gethostid();
682 #endif
683 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)
684 extern int gethostname();
685 #endif

687 #ifndef __GETLOGIN_DEFINED /* Avoid duplicate in stdlib.h */
688 #define __GETLOGIN_DEFINED
689 #ifndef USE_LEGACY_LOGNAME
690 #ifdef PRAGMA_REDEFINE_EXTNAME
691 #pragma redefine_extname getlogin getloginx
692 #else /* PRAGMA_REDEFINE_EXTNAME */
693 extern char *getloginx();
694 #define getlogin getloginx
695 #endif /* PRAGMA_REDEFINE_EXTNAME */
696 #endif /* USE_LEGACY_LOGNAME */
697 extern char *getlogin();
698 #endif /* __GETLOGIN_DEFINED */

700 #if defined(XPG4) || defined(__EXTENSIONS__)
701 extern int getopt();
702 extern char *optarg;
703 extern int opterr, optind, optopt;
704 #if !defined(XPG6) || defined(__EXTENSIONS__)
705 extern char *getpass();
706 #endif
707 #endif /* defined(XPG4) || defined(__EXTENSIONS__) */
708 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)
709 #if !defined(XPG6) || defined(__EXTENSIONS__)
710 extern int getpagesize();
711 #endif
712 extern pid_t getpgid();
713 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG4_2)... */
714 extern pid_t getpid();
715 extern pid_t getppid();
716 extern pid_t getpgrp();
717 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
718 char *gettxt();
719 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
720 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)

```

```

721 extern pid_t getsid();
722 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) */
723 extern uid_t getuid();
724 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
725 extern char *getusershell();
726 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
727 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)
728 extern char *getwd();
729 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG4_2)... */
730 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
731 extern int ioctl();
732 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
733 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
734 extern int isaexec();
735 extern int isetugid();
736 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
737 extern int isatty();
738 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)
739 extern int lchown();
740 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) */
741 extern int link();
742 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
743 extern offset_t llseek();
744 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
745 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || \
746     (defined(LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
747     defined(__EXTENSIONS__)
748 extern int lockf();
749 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG4_2)... */
750 extern off_t lseek();
751 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
752 extern int mincore();
753 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
754 #if !defined(_POSIX_C_SOURCE) || defined(XOPEN_SOURCE) || \
755     defined(__EXTENSIONS__)
756 extern int nice();
757 #endif /* !defined(_POSIX_C_SOURCE) || defined(XOPEN_SOURCE)... */
758 extern long pathconf();
759 extern int pause();
760 extern int pipe();
761 #if !defined(__XOPEN_OR_POSIX) || defined(XPG5) || \
762     (defined(LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
763     defined(__EXTENSIONS__)
764 extern ssize_t pread();
765 #endif
766 #if !defined(LP64) && \
767     (!defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__))
768 extern void profil();
769 extern int ptrace();
770 #endif
771 #if !defined(__XOPEN_OR_POSIX) || \
772     ((_POSIX_C_SOURCE > 2) && !defined(XPG6)) || \
773     defined(__EXTENSIONS__)
774 extern int pthread_atfork();
775 #endif /* !defined(__XOPEN_OR_POSIX) || ((_POSIX_C_SOURCE > 2) ... */
776 #if !defined(__XOPEN_OR_POSIX) || defined(XPG5) || \
777     (defined(LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
778     defined(__EXTENSIONS__)
779 extern ssize_t pwrite();
780 #endif /* !defined(__XOPEN_OR_POSIX) || defined(XPG5) */
781 extern ssize_t read();
782 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
783 /* per RFC 3542; This is also defined in netdb.h */
784 extern int rcmd_af();
785 #endif
786 #if !defined(__XOPEN_OR_POSIX) || defined(XPG4_2) || defined(__EXTENSIONS__)

```

```

787 extern ssize_t readlink();
788 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
789 #if (!defined(__XOPEN_OR_POSIX) || (defined(__XPG3) && !defined(__XPG4))) || \
790     defined(__EXTENSIONS__)
791 extern int rename();
792 #endif /* (!defined(__XOPEN_OR_POSIX) || (defined(__XPG3)... */
793 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
794 extern int resolvepath();
795 /* per RFC 3542; This is also defined in netdb.h */
796 extern int rexec_af();
797 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
798 extern int rmdir();
799 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
800 /* per RFC 3542; This is also defined in netdb.h */
801 extern int rresvport_af();
802 #endif
803 #if !defined(__XOPEN_OR_POSIX) || (defined(__XPG4_2) && !defined(__XPG6)) || \
804     defined(__EXTENSIONS__)
805 extern void *sbrk();
806 #endif /* !defined(__XOPEN_OR_POSIX) || (defined(__XPG4_2)... */
807 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG6) || defined(__EXTENSIONS__)
808 extern int setegid();
809 extern int seteuid();
810 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG6) ... */
811 extern int setgid();
812 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
813 extern int setgroups();
814 extern int sethostname();
815 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
816 extern int setpgid();
817 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
818 extern pid_t setpgid();
819 extern int setregid();
820 extern int setreuid();
821 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
822 extern pid_t setsid();
823 extern int setuid();
824 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
825 extern void setusershell();
826 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
827 extern unsigned sleep();
828 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
829 extern int stime();
830 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
831 #if defined(__XPG4)
832 /* __EXTENSIONS__ makes the SVID Third Edition prototype in stdlib.h visible */
833 extern void swab();
834 #endif /* defined(__XPG4) */
835 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
836 extern int symlink();
837 extern void sync();
838 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
839 #if defined(__XPG5)
840 #ifdef __PRAGMA_REDEFINE_EXTNAME
841 #pragma redefine_extname sysconf __sysconf_xpg5
842 extern long sysconf();
843 #else /* __PRAGMA_REDEFINE_EXTNAME */
844 extern long __sysconf_xpg5();
845 #define sysconf __sysconf_xpg5
846 #endif /* __PRAGMA_REDEFINE_EXTNAME */
847 #endif /* defined(__XPG5) */
848 extern pid_t tcgetpgrp();
849 extern int tcsetpgrp();
850 #if !defined(__XOPEN_OR_POSIX) || \
851     (defined(__LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
852     defined(__EXTENSIONS__)

```

```

853 extern off_t tell();
854 #endif /* !defined(__XOPEN_OR_POSIX)... */
855 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || \
856     (defined(__LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
857     defined(__EXTENSIONS__)
858 extern int truncate();
859 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
860 extern char *ttyname();
861 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
862 extern useconds_t ualarm();
863 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
864 extern int unlink();
865 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
866 extern int usleep();
867 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
868 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
869 extern pid_t vfork();
870 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */
871 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
872 extern void vhangup();
873 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
874 extern ssize_t write();
875 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
876 extern void yield();
877 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

879 #if !defined(__XOPEN_OR_POSIX) || defined(__ATFILE_SOURCE) || \
880     defined(__EXTENSIONS__)
881     /* || defined(__XPG7) */
882 extern int faccessat();
883 extern int fchownat();
884 extern int linkat();
885 extern ssize_t readlinkat();
886 extern int renameat();
887 extern int symlinkat();
888 extern int unlinkat();
889 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__ATFILE_SOURCE)... */
890 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
891 extern int get_nprocs();
892 extern int get_nprocs_conf();
893 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

895 /* transitional large file interface versions */
896 #if defined(__LARGEFILE64_SOURCE) && !((_FILE_OFFSET_BITS == 64) && \
897     !defined(__PRAGMA_REDEFINE_EXTNAME))
898 extern int ftruncate64();
899 extern off64_t lseek64();
900 extern ssize_t pread64();
901 extern ssize_t pwrite64();
902 extern off64_t tell64();
903 extern int truncate64();
904 extern int lockf64();
905 #endif /* __LARGEFILE64_SOURCE */

907 #endif /* __STDC__ */

909 #if !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2) || defined(__EXTENSIONS__)
910 #pragma unknown_control_flow(vfork)
911 #endif /* !defined(__XOPEN_OR_POSIX) || defined(__XPG4_2)... */

913 /*
914 * getlogin_r() & ttyname_r() prototypes are defined here.
915 */

917 /*
918 * Previous releases of Solaris, starting at 2.3, provided definitions of

```



```

919 * various functions as specified in POSIX.1c, Draft 6. For some of these
920 * functions, the final POSIX 1003.1c standard had a different number of
921 * arguments and return values.
922 *
923 * The following segment of this header provides support for the standard
924 * interfaces while supporting applications written under earlier
925 * releases. The application defines appropriate values of the feature
926 * test macros _POSIX_C_SOURCE and _POSIX_PTHREAD_SEMANTICS to indicate
927 * whether it was written to expect the Draft 6 or standard versions of
928 * these interfaces, before including this header. This header then
929 * provides a mapping from the source version of the interface to an
930 * appropriate binary interface. Such mappings permit an application
931 * to be built from libraries and objects which have mixed expectations
932 * of the definitions of these functions.
933 *
934 * For applications using the Draft 6 definitions, the binary symbol is the
935 * same as the source symbol, and no explicit mapping is needed. For the
936 * standard interface, the function func() is mapped to the binary symbol
937 * _posix_func(). The preferred mechanism for the remapping is a compiler
938 * #pragma. If the compiler does not provide such a #pragma, the header file
939 * defines a static function func() which calls the _posix_func() version;
940 * this has to be done instead of #define since POSIX specifies that an
941 * application can #undef the symbol and still be bound to the correct
942 * implementation. Unfortunately, the statics confuse lint so we fallback to
943 * #define in that case.
944 *
945 * NOTE: Support for the Draft 6 definitions is provided for compatibility
946 * only. New applications/libraries should use the standard definitions.
947 */

949 #if defined(__EXTENSIONS__) || defined(_REENTRANT) || \
950 !defined(__XOPEN_OR_POSIX) || (_POSIX_C_SOURCE - 0 >= 199506L) || \
951 defined(_POSIX_PTHREAD_SEMANTICS)

953 #if defined(__STDC__)

955 #if (_POSIX_C_SOURCE - 0 >= 199506L) || defined(_POSIX_PTHREAD_SEMANTICS)

957 #ifndef __USE_LEGACY_LOGNAME__
958 #ifdef __PRAGMA_REDEFINE_EXTNAME
959 #pragma redefine_extname getlogin_r __posix_getloginx_r
960 extern int getlogin_r(char *, int);
961 #else /* __PRAGMA_REDEFINE_EXTNAME */
962 extern int __posix_getloginx_r(char *, int);
963 #define getlogin_r __posix_getloginx_r
964 #endif /* __PRAGMA_REDEFINE_EXTNAME */
965 #else /* __USE_LEGACY_LOGNAME__ */
966 #ifdef __PRAGMA_REDEFINE_EXTNAME
967 #pragma redefine_extname getlogin_r __posix_getlogin_r
931 #pragma redefine_extname ttyname_r __posix_ttyname_r
968 extern int getlogin_r(char *, int);
933 extern int ttyname_r(int, char *, size_t);
969 #else /* __PRAGMA_REDEFINE_EXTNAME */

970 extern int __posix_getlogin_r(char *, int);
937 extern int __posix_ttyname_r(int, char *, size_t);

972 #ifdef __lint

974 #define getlogin_r __posix_getlogin_r
942 #define ttyname_r __posix_ttyname_r

976 #else /* !__lint */

978 static int
979 getlogin_r(char *__name, int __len)

```

```

980 {
981     return (__posix_getlogin_r(__name, __len));
982 }
983 static int

985 #endif /* !__lint */
986 #endif /* __PRAGMA_REDEFINE_EXTNAME */
987 #endif /* __USE_LEGACY_LOGNAME__ */

989 #ifdef __PRAGMA_REDEFINE_EXTNAME
990 #pragma redefine_extname ttyname_r __posix_ttyname_r
991 extern int ttyname_r(int, char *, size_t);
992 #else /* __PRAGMA_REDEFINE_EXTNAME */
993 extern int __posix_ttyname_r(int, char *, size_t);

995 #ifdef __lint

997 #define ttyname_r __posix_ttyname_r

999 #else /* !__lint */

1001 static int
1002 ttyname_r(int __fildev, char *__buf, size_t __size)
1003 {
1004     return (__posix_ttyname_r(__fildev, __buf, __size));
1005 }

1007 #endif /* !__lint */
1008 #endif /* __PRAGMA_REDEFINE_EXTNAME */

1010 #else /* (_POSIX_C_SOURCE - 0 >= 199506L) || ... */

1012 #ifndef __USE_LEGACY_LOGNAME__
1013 #ifdef __PRAGMA_REDEFINE_EXTNAME
1014 #pragma redefine_extname getlogin_r getloginx_r
1015 #else /* __PRAGMA_REDEFINE_EXTNAME */
1016 extern char *getloginx_r(char *, int);
1017 #define getlogin_r getloginx_r
1018 #endif /* __PRAGMA_REDEFINE_EXTNAME */
1019 #endif /* __USE_LEGACY_LOGNAME__ */
1020 extern char *getlogin_r(char *, int);

1022 extern char *ttyname_r(int, char *, int);

1024 #endif /* (_POSIX_C_SOURCE - 0 >= 199506L) || ... */

1026 #else /* __STDC__ */

1028 #if (_POSIX_C_SOURCE - 0 >= 199506L) || defined(_POSIX_PTHREAD_SEMANTICS)

1030 #ifndef __USE_LEGACY_LOGNAME__
1031 #ifdef __PRAGMA_REDEFINE_EXTNAME
1032 #pragma redefine_extname getlogin_r __posix_getloginx_r
1033 extern int getlogin_r();
1034 #else /* __PRAGMA_REDEFINE_EXTNAME */
1035 extern int __posix_getloginx_r();
1036 #define getlogin_r __posix_getloginx_r
1037 #endif /* __PRAGMA_REDEFINE_EXTNAME */
1038 #else /* __USE_LEGACY_LOGNAME__ */
1039 #ifdef __PRAGMA_REDEFINE_EXTNAME
1040 #pragma redefine_extname getlogin_r __posix_getlogin_r
973 #pragma redefine_extname ttyname_r __posix_ttyname_r
1041 extern int getlogin_r();
975 extern int ttyname_r();
1042 #else /* __PRAGMA_REDEFINE_EXTNAME */

```

```

1043 extern int __posix_getlogin_r();
1044 extern int __posix_ttyname_r();

1045 #ifdef __lint

1047 #define getlogin_r      __posix_getlogin_r
1048 #define ttyname_r      __posix_ttyname_r

1049 #else /* !__lint */

1051 static int
1052 getlogin_r(__name, __len)
1053     char *__name;
1054     int __len;
1055 {
1056     return (__posix_getlogin_r(__name, __len));
1057 }
1058 #endif /* !__lint */
1059 #endif /* __PRAGMA_REDEFINE_EXTNAME */
1060 #endif /* __USE_LEGACY_LOGNAME */

1062 #ifdef __PRAGMA_REDEFINE_EXTNAME
1063 #pragma redefine_extname ttyname_r __posix_ttyname_r
1064 extern int ttyname_r();
1065 #else /* __PRAGMA_REDEFINE_EXTNAME */

1067 extern int __posix_ttyname_r();

1069 #ifdef __lint

1071 #define ttyname_r      __posix_ttyname_r

1073 #else /* !__lint */

    995 static int
1075 ttyname_r(__fildes, __buf, __size)
1076     int __fildes;
1077     char *__buf;
1078     size_t __size;
1079 {
1080     return (__posix_ttyname_r(__fildes, __buf, __size));
1081 }
1082 #endif /* !__lint */
1083 #endif /* __PRAGMA_REDEFINE_EXTNAME */

1085 #else /* (_POSIX_C_SOURCE - 0 >= 199506L) || ... */

1087 #ifndef __USE_LEGACY_LOGNAME
1088 #ifdef __PRAGMA_REDEFINE_EXTNAME
1089 #pragma redefine_extname getlogin_r      getloginx_r
1090 #else /* __PRAGMA_REDEFINE_EXTNAME */
1091 extern char *getloginx_r();
1092 #define getlogin_r      getloginx_r
1093 #endif /* __PRAGMA_REDEFINE_EXTNAME */
1094 #endif /* __USE_LEGACY_LOGNAME */
1095 extern char *getlogin_r();

1097 extern char *ttyname_r();

1099 #endif /* (_POSIX_C_SOURCE - 0 >= 199506L) || ... */

1101 #endif /* __STDC__ */

1103 #endif /* defined(__EXTENSIONS__) || defined(_REENTRANT)... */

1105 #ifdef __cplusplus

```

```

1106 }
    unchanged_portion_omitted

```

new/usr/src/lib/libc/port/gen/getlogin.c

1

```
*****
4669 Sat Jun 1 10:02:50 2013
new/usr/src/lib/libc/port/gen/getlogin.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
25  * Copyright 2008 Sun Microsystems, Inc. All rights reserved.
26  * Use is subject to license terms.
27  */

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

32 #pragma weak _getlogin = getloginx
33 #pragma weak _getlogin_r = getloginx_r
30 #pragma ident      "%Z%M% %I%      %E% SMI"

32 #pragma weak _getlogin = getlogin
33 #pragma weak _getlogin_r = getlogin_r

35 #include "lint.h"
36 #include <sys/types.h>
37 #include <sys/stat.h>
38 #include <fcntl.h>
39 #include <string.h>
40 #include <stdlib.h>
41 #include <limits.h>
42 #include "utmpx.h"
43 #include <unistd.h>
44 #include <errno.h>
45 #include <thread.h>
46 #include <synch.h>
47 #include <mtlib.h>
48 #include "tsd.h"

50 /* Revert the renames done in unistd.h */
51 #ifndef __PRAGMA_REDEFINE_EXTNAME
52 #pragma redefine_extname      getlogint      getlogin
53 #pragma redefine_extname      getlogint_r    getlogin_r
54 #pragma redefine_extname      __posix_getlogint_r    __posix_getlogin_r
55 #else /* __PRAGMA_REDEFINE_EXTNAME */
56 #ifdef getlogin
```

new/usr/src/lib/libc/port/gen/getlogin.c

2

```
57 #undef      getlogin
58 #endif /* getlogin */
59 #ifdef getlogin_r
60 #undef      getlogin_r
61 #endif /* getlogin_r */
62 #ifdef __posix_getlogin_r
63 #undef      __posix_getlogin_r
64 #endif /* __posix_getlogin_r */
65 #define getlogint      getlogin
66 #define getlogint_r    getlogin_r
67 #define __posix_getlogint_r    __posix_getlogin_r
68 #endif /* __PRAGMA_REDEFINE_EXTNAME */

70 /*
71  * Use the full length of a login name.
72  * The utmpx interface provides for a 32 character login name.
73  * XXX - _POSIX_LOGIN_NAME_MAX limits the length of a login name. The utmpx
74  * interface provides for a 32 character login name, but for the sake of
75  * compatibility, we are still using the old utmp-imposed limit.
76  */
77 #define NMAX      (sizeof (((struct utmpx *)0)->ut_user))

76 /*
77  * Common function
78  * POSIX.1c Draft-6 version of the function getlogin_r.
79  * It was implemented by Solaris 2.3.
80  */
81 static char *
82 getl_r_common(char *answer, size_t namelen, size_t maxlen)
83 char *
84 getlogin_r(char *answer, int namelen)
85 {
86     int          uf;
87     off64_t      me;
88     struct futmpx ubuf;

89     if (namelen < _POSIX_LOGIN_NAME_MAX) {
90         errno = ERANGE;
91         return (NULL);
92     }

93     if ((me = (off64_t)ttyslot()) < 0)
94         return (NULL);
95     if ((uf = open64(UTMPX_FILE, 0)) < 0)
96         return (NULL);
97     (void) lseek64(uf, me * sizeof (ubuf), SEEK_SET);
98     if (read(uf, &ubuf, sizeof (ubuf)) != sizeof (ubuf)) {
99         (void) close(uf);
100        return (NULL);
101    }
102    (void) close(uf);
103    if (ubuf.ut_user[0] == '\0')
104        return (NULL);

105    /* Insufficient buffer size */
106    if (namelen < strlen(&ubuf.ut_user[0], maxlen)) {
107        errno = ERANGE;
108        return (NULL);
109    }
110    (void) strncpy(&answer[0], &ubuf.ut_user[0], maxlen);
111    answer[maxlen] = '\0';
112    (void) strncpy(&answer[0], &ubuf.ut_user[0],
113        _POSIX_LOGIN_NAME_MAX - 1);
114    answer[_POSIX_LOGIN_NAME_MAX - 1] = '\0';
115    return (&answer[0]);
116 }
```

```

109 /*
110 * POSIX.1c Draft-6 version of the function getlogin_r.
111 * It was implemented by Solaris 2.3.
112 */
113 char *
114 getlogint_r(char *answer, int namelen)
115 {
116     return (getl_r_common(answer, (size_t)namelen, LOGNAME_MAX_TRAD));
117 }

119 /*
120 * POSIX.1c standard version of the function getlogin_r.
121 * User gets it via static getlogin_r from the header file.
122 */
123 int
124 __posix_getlogint_r(char *name, int namelen)
125     __posix_getlogin_r(char *name, int namelen)
126 {
127     int nerrno = 0;
128     int oerrno = errno;

129     errno = 0;
130     if (getl_r_common(name, (size_t)namelen, LOGNAME_MAX_TRAD) == NULL) {
131         if (getlogin_r(name, namelen) == NULL) {
132             if (errno == 0)
133                 nerrno = EINVAL;
134             else
135                 nerrno = errno;
136         }
137         errno = oerrno;
138         return (nerrno);
139     }

140 char *
141 getlogint(void)
142     getlogin(void)
143 {
144     char *answer = tsdalloc(_T_LOGIN, LOGIN_NAME_MAX_TRAD, NULL);
145     char *answer = tsdalloc(_T_LOGIN, _POSIX_LOGIN_NAME_MAX, NULL);

146     if (answer == NULL)
147         return (NULL);
148     return (getl_r_common(answer, LOGIN_NAME_MAX_TRAD, LOGNAME_MAX_TRAD));
149 }

150 /*
151 * POSIX.1c Draft-6 version of the function getlogin_r.
152 * It was implemented by Solaris 2.3.
153 * For extended login names, selected by redefine_extname in unistd.h.
154 */
155 char *
156 getloginx_r(char *answer, int namelen)
157 {
158     return (getl_r_common(answer, (size_t)namelen, NMAX));
159 }

161 /*
162 * POSIX.1c standard version of the function getlogin_r.
163 * User gets it via static getlogin_r from the header file.
164 * For extended login names, selected by redefine_extname in unistd.h.
165 */
166 int
167 __posix_getloginx_r(char *name, int namelen)
168 {
169     int nerrno = 0;

```

```

170     int oerrno = errno;

172     errno = 0;
173     if (getl_r_common(name, (size_t)namelen, NMAX) == NULL) {
174         if (errno == 0)
175             nerrno = EINVAL;
176         else
177             nerrno = errno;
178     }
179     errno = oerrno;
180     return (nerrno);
181 }

183 /*
184 * For extended login names, selected by redefine_extname in unistd.h.
185 */
186 char *
187 getloginx(void)
188 {
189     char *answer = tsdalloc(_T_LOGIN, LOGIN_NAME_MAX, NULL);

191     if (answer == NULL)
192         return (NULL);
193     return (getl_r_common(answer, LOGIN_NAME_MAX, NMAX));
194     return (getlogin_r(answer, _POSIX_LOGIN_NAME_MAX));
195 }

```

\_\_\_\_\_unchanged\_portion\_omitted\_\_\_\_\_

```

*****
12091 Sat Jun 1 10:02:50 2013
new/usr/src/lib/libc/port/gen/sysconf.c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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  *
25  * Copyright 2009 Sun Microsystems, Inc. All rights reserved.
26  * Use is subject to license terms.
27  */

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

32 /* sysconf(3C) - returns system configuration information */

34 #pragma weak _sysconf = sysconf

36 #include "lint.h"
37 #include <mtlib.h>
38 #include <sys/types.h>
39 #include <unistd.h>
40 #include <sys/sysconfig.h>
41 #include <limits.h>
42 #include <time.h>
43 #include <errno.h>
44 #include <nss_dbdefs.h>
45 #include <thread.h>
46 #include <xti.h>
47 #include "libc.h"
48 #include "xpg6.h"

50 /* from nss_common.c */
51 extern size_t _nss_get_bufsizes(int);

53 long
54 sysconf(int name)
55 {
56     static int _pagesize = 0;
57     static int _hz = 0;
58     static pid_t _maxpid = 0;
59     static int _stackprot = 0;
60     static int _ngroups_max;

```

```

61     extern int __xpg4;

63     switch (name) {
64     default:
65         errno = EINVAL;
66         return (-1L);

68     case _SC_ARG_MAX:
69         return ((long)ARG_MAX);

71     case _SC_CLK_TCK:
72         if (_hz <= 0)
73             _hz = _sysconfig(_CONFIG_CLK_TCK);
74         return (_hz);

76     case _SC_JOB_CONTROL:
77         return ((long)_POSIX_JOB_CONTROL);

79     case _SC_SAVED_IDS:
80         return ((long)_POSIX_SAVED_IDS);

82     case _SC_CHILD_MAX:
83         return (_sysconfig(_CONFIG_CHILD_MAX));

85     case _SC_NGROUPS_MAX:
86         if (_ngroups_max <= 0)
87             _ngroups_max = _sysconfig(_CONFIG_NGROUPS);
88         return (_ngroups_max);

90     case _SC_OPEN_MAX:
91         return (_sysconfig(_CONFIG_OPEN_FILES));

93     case _SC_VERSION:
94         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
95             return (200112L);
96         else
97             return (199506L);

99     case _SC_PAGESIZE:
100         if (_pagesize <= 0)
101             _pagesize = _sysconfig(_CONFIG_PAGESIZE);
102         return (_pagesize);

104     case _SC_XOPEN_VERSION:
105         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
106             return (600L);
107         else if (__xpg4 == 0)
108             return (_sysconfig(_CONFIG_XOPEN_VER));
109         else
110             return (4L);

112     case _SC_XOPEN_XCU_VERSION:
113         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
114             return (600L);
115         else
116             return (4L);

118     /*
119     * old value for pre XPG5 conformant systems to match
120     * getpass() length.
121     * XPG5 special cased with __sysconf_xpg5()
122     * new value for default and modern XPG systems.
123     */
124     case _SC_PASS_MAX:
125         if ((__xpg4 == 1) &&
126             (!(__xpg6 & _C99SUSv3_XPG6_sysconf_version)))

```

```

127         return ((long)_PASS_MAX_XPG);
128     else
129         return ((long)_PASS_MAX);

131     case _SC_LOGNAME_MAX:
132         return ((long)LOGNAME_MAX);

134     case _SC_STREAM_MAX:
135         return (_sysconf(_CONFIG_OPEN_FILES));

137     case _SC_TZNAME_MAX:
138         return (-1L);

140     case _SC_NPROCESSORS_CONF:
141         return (_sysconf(_CONFIG_NPROC_CONF));

143     case _SC_NPROCESSORS_ONLN:
144         return (_sysconf(_CONFIG_NPROC_ONLN));

146     case _SC_NPROCESSORS_MAX:
147         return (_sysconf(_CONFIG_NPROC_MAX));

149     case _SC_STACK_PROT:
150         if (_stackprot == 0)
151             _stackprot = _sysconf(_CONFIG_STACK_PROT);
152         return (_stackprot);

154     /* POSIX.4 names */

156     /*
157     * Each of the following also have _POSIX_* symbols
158     * defined in <unistd.h>. Values here should align
159     * with values in the header. Up until the SUSv3 standard
160     * we defined these simply as 1. With the introduction
161     * of the new revision, these were changed to 200112L.
162     * The standard allows us to change the value, however,
163     * we have kept both values in case application programs
164     * are relying on the previous value even though an
165     * application doing so is technically wrong.
166     */
167     case _SC_ASYNCHRONOUS_IO:
168     case _SC_FSYNC:
169     case _SC_MAPPED_FILES:
170     case _SC_MEMLOCK:
171     case _SC_MEMLOCK_RANGE:
172     case _SC_MEMORY_PROTECTION:
173     case _SC_MESSAGE_PASSING:
174     case _SC_PRIORITY_SCHEDULING:
175     case _SC_REALTIME_SIGNALS:
176     case _SC_SEMAPHORES:
177     case _SC_SHARED_MEMORY_OBJECTS:
178     case _SC_SYNCHRONIZED_IO:
179     case _SC_TIMERS:
180         if (__xpg6 & _C99SUSv3_mode_ON)
181             return (200112L);
182         else
183             return (1L);

185     case _SC_PRIORITIZED_IO:
186 #ifdef _POSIX_PRIORITIZED_IO
187         return (1L);
188 #else
189         return (-1L);
190 #endif

192     case _SC_AIO_LISTIO_MAX:

```

```

193         return (_sysconf(_CONFIG_AIO_LISTIO_MAX));

195     case _SC_AIO_MAX:
196         return (_sysconf(_CONFIG_AIO_MAX));

198     case _SC_AIO_PRIO_DELTA_MAX:
199         return (_sysconf(_CONFIG_AIO_PRIO_DELTA_MAX));

201     case _SC_DELAYTIMER_MAX:
202         return (_sysconf(_CONFIG_DELAYTIMER_MAX));

204     case _SC_MQ_OPEN_MAX:
205         return (_sysconf(_CONFIG_MQ_OPEN_MAX));

207     case _SC_MQ_PRIO_MAX:
208         return (_sysconf(_CONFIG_MQ_PRIO_MAX));

210     case _SC_RTSIG_MAX:
211         return (_sysconf(_CONFIG_RTSIG_MAX));

213     case _SC_SEM_NSEMS_MAX:
214         return (_sysconf(_CONFIG_SEM_NSEMS_MAX));

216     case _SC_SEM_VALUE_MAX:
217         return (_sysconf(_CONFIG_SEM_VALUE_MAX));

219     case _SC_SIGQUEUE_MAX:
220         return (_sysconf(_CONFIG_SIGQUEUE_MAX));

222     case _SC_SIGRT_MAX:
223         return (_sysconf(_CONFIG_SIGRT_MAX));

225     case _SC_SIGRT_MIN:
226         return (_sysconf(_CONFIG_SIGRT_MIN));

228     case _SC_TIMER_MAX:
229         return (_sysconf(_CONFIG_TIMER_MAX));

231     case _SC_PHYS_PAGES:
232         return (_sysconf(_CONFIG_PHYS_PAGES));

234     case _SC_AVPHYS_PAGES:
235         return (_sysconf(_CONFIG_AVPHYS_PAGES));

237     /* XPG4/POSIX.1-1990/POSIX.2-1992 names */
238     case _SC_2_C_BIND:
239         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
240             return (200112L);
241         else
242             return (1L);

244     case _SC_2_CHAR_TERM:
245         return ((long)_POSIX2_CHAR_TERM);

247     case _SC_2_C_DEV:
248         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
249             return (200112L);
250         else
251             return (1L);

253     case _SC_2_C_VERSION:
254         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
255             return (200112L);
256         else
257             return (199209L);

```

```

259     case _SC_2_FORT_DEV:
260         return (-1L);

262     case _SC_2_FORT_RUN:
263         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
264             return (200112L);
265         else
266             return (1L);

268     case _SC_2_LOCALEDEF:
269         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
270             return (200112L);
271         else
272             return (1L);

274     case _SC_2_SW_DEV:
275         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
276             return (200112L);
277         else
278             return (1L);

280     case _SC_2_UPE:
281         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
282             return (200112L);
283         else
284             return (1L);

286     case _SC_2_VERSION:
287         if (__xpg6 & _C99SUSv3_XPG6_sysconf_version)
288             return (200112L);
289         else
290             return (199209L);

292     case _SC_BC_BASE_MAX:
293         return ((long)BC_BASE_MAX);

295     case _SC_BC_DIM_MAX:
296         return ((long)BC_DIM_MAX);

298     case _SC_BC_SCALE_MAX:
299         return ((long)BC_SCALE_MAX);

301     case _SC_BC_STRING_MAX:
302         return ((long)BC_STRING_MAX);

304     case _SC_COLL_WEIGHTS_MAX:
305         return ((long)COLL_WEIGHTS_MAX);

307     case _SC_EXPR_NEST_MAX:
308         return ((long)EXPR_NEST_MAX);

310     case _SC_LINE_MAX:
311         return ((long)LINE_MAX);

313     case _SC_RE_DUP_MAX:
314         return ((long)RE_DUP_MAX);

316     case _SC_XOPEN_CRYPT:
317         return (1L);

319     case _SC_XOPEN_ENH_I18N:
320         return ((long)_XOPEN_ENH_I18N);

322     case _SC_XOPEN_SHM:
323         return ((long)_XOPEN_SHM);

```

```

325         /* XPG4v2 (SUS) names */
326     case _SC_XOPEN_UNIX:
327         return (1L);

329     case _SC_XOPEN_LEGACY:
330         return (1L);

332     case _SC_ATEXIT_MAX:
333         return (-1L);

335     case _SC_IOV_MAX:
336         return ((long)IOV_MAX);

338     case _SC_T_IOV_MAX:
339         return ((long)T_IOV_MAX);

341         /* XPG5 (SUSv2) names */
342     case _SC_XOPEN_REALTIME:
343         return (1L);

345     case _SC_XOPEN_REALTIME_THREADS:
346 #if defined(_POSIX_THREAD_PRIORITY_SCHEDULING) && \
347     defined(_POSIX_THREAD_PRIO_INHERIT) && \
348     defined(_POSIX_THREAD_PRIO_PROTECT)
349         return (1L);
350 #else
351         return (-1L);
352 #endif

354     case _SC_XBS5_ILP32_OFF32:
355         return (1L);

357     case _SC_XBS5_ILP32_OFFBIG:
358         return (1L);

360     case _SC_XBS5_LP64_OFF64:
361         return (1L);

363     case _SC_XBS5_LPBIG_OFFBIG:
364         return (1L);

366         /* POSIX.1c names */
367     case _SC_THREAD_DESTRUCTOR_ITERATIONS:
368         return (-1L);

370     case _SC_GETGR_R_SIZE_MAX:
371         return ((long)_nss_get_bufsizes(_SC_GETGR_R_SIZE_MAX));

373     case _SC_GETPW_R_SIZE_MAX:
374         return ((long)_NSS_BUFLEN_PASSWD);

376     case _SC_LOGIN_NAME_MAX:
377         return ((long)(LOGIN_NAME_MAX));
375         return ((long)(LOGNAME_MAX + 1));

379     case _SC_THREAD_KEYS_MAX:
380         return (-1L);

382     case _SC_THREAD_STACK_MIN:
383         return ((long)thr_min_stack());

385     case _SC_THREAD_THREADS_MAX:
386         return (-1L);

388     case _SC_TTY_NAME_MAX:
389         return ((long)TTYNAME_MAX);

```

```

391     case _SC_BARRIERS:
392         return ((long)_POSIX_BARRIERS);

394     case _SC_CLOCK_SELECTION:
395         return ((long)_POSIX_CLOCK_SELECTION);

397     case _SC_MONOTONIC_CLOCK:
398         return ((long)_POSIX_MONOTONIC_CLOCK);

400     case _SC_SPAWN:
401         return ((long)_POSIX_SPAWN);

403     case _SC_SPIN_LOCKS:
404         return ((long)_POSIX_SPIN_LOCKS);

406     case _SC_THREADS:
407     case _SC_THREAD_ATTR_STACKADDR:
408     case _SC_THREAD_ATTR_STACKSIZE:
409     case _SC_THREAD_PRIORITY_SCHEDULING:
410     case _SC_THREAD_PRIO_INHERIT:
411     case _SC_THREAD_PRIO_PROTECT:
412     case _SC_THREAD_PROCESS_SHARED:
413     case _SC_THREAD_SAFE_FUNCTIONS:
414         if (__xpg6 & _C99SUSv3_mode_ON)
415             return (200112L);
416         else
417             return (1L);

419     case _SC_TIMEOUTS:
420         return ((long)_POSIX_TIMEOUTS);

422     /* 1216676 - cache info */
423     case _SC_COHER_BLKSZ:
424         return (_sysconf(_CONFIG_COHERENCY));

426     case _SC_SPLIT_CACHE:
427         return (_sysconf(_CONFIG_SPLIT_CACHE));

429     case _SC_ICACHE_SZ:
430         return (_sysconf(_CONFIG_ICACHESZ));

432     case _SC_DCACHE_SZ:
433         return (_sysconf(_CONFIG_DCACHESZ));

435     case _SC_ICACHE_LINESZ:
436         return (_sysconf(_CONFIG_ICACHELINESZ));

438     case _SC_DCACHE_LINESZ:
439         return (_sysconf(_CONFIG_DCACHELINESZ));

441     case _SC_ICACHE_BLKSZ:
442         return (_sysconf(_CONFIG_ICACHEBLKSZ));

444     case _SC_DCACHE_BLKSZ:
445         return (_sysconf(_CONFIG_DCACHEBLKSZ));

447     case _SC_DCACHE_TBLKSZ:
448         return (_sysconf(_CONFIG_DCACHE_TBLKSZ));

450     case _SC_ICACHE_ASSOC:
451         return (_sysconf(_CONFIG_ICACHE_ASSOC));

453     case _SC_DCACHE_ASSOC:
454         return (_sysconf(_CONFIG_DCACHE_ASSOC));

```

```

456     case _SC_MAXPID:
457         if (_maxpid <= 0)
458             _maxpid = _sysconf(_CONFIG_MAXPID);
459         return (_maxpid);

461     case _SC_CPUID_MAX:
462         return (_sysconf(_CONFIG_CPUID_MAX));

464     case _SC_EPHID_MAX:
465         return (_sysconf(_CONFIG_EPHID_MAX));

467     /* UNIX 03 names - XPG6/SUSv3/POSIX.1-2001 */

469     case _SC_REGEXP:
470         return ((long)_POSIX_REGEXP);

472     case _SC_SHELL:
473         return ((long)_POSIX_SHELL);

475     case _SC_ADVISORY_INFO:
476         return ((long)_POSIX_ADVISORY_INFO);

478     case _SC_HOST_NAME_MAX:
479         return ((long)_POSIX_HOST_NAME_MAX);

481     case _SC_READER_WRITER_LOCKS:
482         return ((long)_POSIX_READER_WRITER_LOCKS);

484     case _SC_IPV6:
485         return ((long)_POSIX_IPV6);

487     case _SC_RAW_SOCKETS:
488         return ((long)_POSIX_RAW_SOCKETS);

490     case _SC_XOPEN_STREAMS:
491         return ((long)_XOPEN_STREAMS);

493     case _SC_SYMLINK_MAX:
494         return (_sysconf(_CONFIG_SYMLINK_MAX));

496     case _SC_V6_ILP32_OFF32:
497         return (1L);

499     case _SC_V6_ILP32_OFFBIG:
500         return (1L);

502     case _SC_V6_LP64_OFF64:
503         return (1L);

505     case _SC_V6_LP64_OFFBIG:
506         return (1L);

508     /* Unsupported UNIX 03 options */
509     case _SC_2_PBS:
510     case _SC_2_PBS_ACCOUNTING:
511     case _SC_2_PBS_CHECKPOINT:
512     case _SC_2_PBS_LOCATE:
513     case _SC_2_PBS_MESSAGE:
514     case _SC_2_PBS_TRACK:
515     case _SC_CPUTIME:
516     case _SC_SPORADIC_SERVER:
517     case _SC_SS_REPL_MAX:
518     case _SC_THREAD_CPUTIME:
519     case _SC_THREAD_SPORADIC_SERVER:
520     case _SC_TRACE:
521     case _SC_TRACE_EVENT_FILTER:

```



```
522         case _SC_TRACE_EVENT_NAME_MAX:
523         case _SC_TRACE_INHERIT:
524         case _SC_TRACE_LOG:
525         case _SC_TRACE_NAME_MAX:
526         case _SC_TRACE_SYS_MAX:
527         case _SC_TRACE_USER_EVENT_MAX:
528         case _SC_TYPED_MEMORY_OBJECTS:
529             return (-1L);
530     }
531 }
unchanged_portion_omitted
```

```

*****
45694 Sat Jun 1 10:02:51 2013
new/usr/src/lib/libc/port/llib-1c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
24 * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
25 * Copyright (c) 2013 Gary Mills
26 */

28 /* LINTLIBRARY */
29 /* PROTLIB1 */

31 #define __EXTENSIONS__

33 #include <aio.h>
34 #include <alloca.h>
35 #include <attr.h>
36 #include <atomic.h>
37 #include <ctype.h>
38 #include <deflt.h>
39 #include <dirent.h>
40 #include <dlfcn.h>
41 #include <door.h>
42 #include <err.h>
43 #include <sys/errno.h>
44 #include <euc.h>
45 #include <fcntl.h>
46 #include <float.h>
47 #include <fmtmsg.h>
48 #include <fnmatch.h>
49 #include <ftw.h>
50 #include <glob.h>
51 #include <getwidth.h>
52 #include <grp.h>
53 #include <iconv.h>
54 #include <langinfo.h>
55 #include <libgen.h>
56 #include <libw.h>
57 #include <locale.h>
58 #include <memory.h>
59 #include <mon.h>
60 #include <mqueue.h>

```

```

61 #include <nan.h>
62 #include <ndbm.h>
63 #include <limits.h>
64 #include <nl_types.h>
65 #include <poll.h>
66 #include <project.h>
67 #include <priv.h>
68 #include <pwd.h>
69 #include <rctl.h>
70 #include <regex.h>
71 #include <rpcsvc/ypclnt.h>
72 #include <sched.h>
73 #include <search.h>
74 #include <semaphore.h>
75 #include <setjmp.h>
76 #include <shadow.h>
77 #include <siginfo.h>
78 #include <signal.h>
79 #include <stdarg.h>
80 #include <ucred.h>
81 #include <sys/ucred.h>
82 #include <unistd.h>
83 #include <ulimit.h>
84 #include <utime.h>
85 #include <stddef.h>
86 #include <stdio.h>
87 #include <stdlib.h>
88 #include <string.h>
89 #include <stropts.h>
90 #include <synch.h>
91 #include <sys/acctctl.h>
92 #include <sys/acl.h>
93 #include <sys/asynch.h>
94 #include <sys/byteorder.h>
95 #include <sys/cladm.h>
96 #include <sys/corectl.h>
97 #include <sys/dl.h>
98 #include <sys/exacct.h>
99 #include <sys/fcntl.h>
100 #include <sys/file.h>
101 #include <sys/fs/namenode.h>
102 #include <sys/instance.h>
103 #include <sys/ipc.h>
104 #include <sys/lwp.h>
105 #include <sys/mkdev.h>
106 #include <sys/mman.h>
107 #include <sys/mnttab.h>
108 #include <sys/mount.h>
109 #include <sys/msg.h>
110 #include <sys/param.h>
111 #include <sys/priocntl.h>
112 #include <sys/procset.h>
113 #include <sys/processor.h>
114 #include <sys/pset.h>
115 #include <sys/rctl_impl.h>
116 #include <sys/sem.h>
117 #include <sys/shm.h>
118 #include <sys/sid.h>
119 #include <sys/signal.h>
120 #include <sys/stat.h>
121 #include <sys/statvfs.h>
122 #include <sys/strlog.h>
123 #include <sys/stropts.h>
124 #include <sys/syscall.h>
125 #include <sys/sysconfig.h>
126 #include <sys/syslog.h>

```

```

127 #include <sys/systeminfo.h>
128 #include <sys/task.h>
129 #include <sys/termio.h>
130 #include <sys/termios.h>
131 #include <sys/u8_textprep.h>
132 #include <sys/time.h>
133 #include <sys/timeb.h>
134 #include <sys/times.h>
135 #include <sys/types.h>
136 #include <sys/uadmin.h>
137 #include <sys/utsname.h>
138 #include <sys/vfstab.h>
139 #include <sys/sendfile.h>
140 #include <sys/zone.h>
141 #include <termio.h>
142 #include <time.h>
143 #include <tzfile.h>
144 #include <ucontext.h>
145 #include <utmpx.h>
146 #include <values.h>
147 #include <wait.h>
148 #include <wchar.h>
149 #include <wctype.h>
150 #include <wider.h>
151 #include <wordexp.h>
152 #include <thread.h>
153 #include <pthread.h>
154 #include <schedctl.h>
155 #include <zone.h>
156 #include <port.h>
157 #include <spawn.h>
158 #include <inttypes.h>
159 #include <getopt.h>
160 #include <stdio_ext.h>
161 #if defined(__i386)
162 #include <sys/sysi86.h>
163 #endif
164 #if defined(__amd64)
165 #include <stack_unwind.h>
166 #endif

168 /*
169  * This really comes from the crt*.s startup modules.
170  */
171 char **environ;

173 /*
174  * This is a GNU/Linux/BSD compatibility interface,
175  * not declared in any header file.
176  */
177 const char *__progname;

179 /*
180  * POSIX versions of standard libc routines; these aren't extracted
181  * from the headers above since we cannot #define _POSIX_C_SOURCE.
182  */
183 int __posix_readdir_r(DIR * _RESTRICT_KYWD, struct dirent * _RESTRICT_KYWD,
184 struct dirent ** _RESTRICT_KYWD);
185 int __posix_getgrgid_r(gid_t, struct group *, char *, size_t, struct group **);
186 int __posix_getgrnam_r(const char *, struct group *, char *, size_t,
187 struct group **);
188 int __posix_getpwuid_r(uid_t, struct passwd *, char *, size_t,
189 struct passwd **);
190 int __posix_getpwnam_r(const char *, struct passwd *, char *, size_t,
191 struct passwd **);
192 int __posix_sigwait(const sigset_t * _RESTRICT_KYWD, int * _RESTRICT_KYWD);

```

```

193 char *__posix_asctime_r(const struct tm * _RESTRICT_KYWD, char * _RESTRICT_KYWD);
194 char *__posix_ctime_r(const time_t *, char *);
195 int __posix_ttyname_r(int, char *, size_t);
196 int __posix_getlogin_r(char *, int);
197 int __posix_getloginx_r(char *, int);

199 /*
200  * XPG4 versions of standard libc routines; these aren't extracted
201  * from the headers above since we cannot #define _XPG4_2.
202  */
203 int __xpg4_putmsg(int, const struct strbuf *, const struct strbuf *, int);
204 int __xpg4_putpmsg(int, const struct strbuf *, const struct strbuf *, int, int);

206 /*
207  * These aren't extracted from the headers above because:
208  * - We cannot #define _STRPTIME_DONTZERO
209  * - We cannot #define _XPG5
210  */
211 char *__strptime_dontzero(const char *, const char *, struct tm *);
212 long __sysconf_xpg5(int);
213 wchar_t *__wcstok_xpg5(wchar_t * _RESTRICT_KYWD,
214 const wchar_t * _RESTRICT_KYWD, wchar_t ** _RESTRICT_KYWD);
215 size_t __wcsftime_xpg5(wchar_t * _RESTRICT_KYWD, size_t,
216 const wchar_t * _RESTRICT_KYWD, const struct tm * _RESTRICT_KYWD);
217 wint_t __fgetwc_xpg5(__FILE *);
218 wint_t __getwc_xpg5(__FILE *);
219 wint_t __getwchar_xpg5(void);
220 wint_t __fputwc_xpg5(wint_t, __FILE *);
221 wint_t __putwc_xpg5(wint_t, __FILE *);
222 wint_t __putwchar_xpg5(wint_t);
223 wchar_t *__fgetws_xpg5(wchar_t * _RESTRICT_KYWD, int, __FILE * _RESTRICT_KYWD);
224 int __fputws_xpg5(const wchar_t * _RESTRICT_KYWD, __FILE * _RESTRICT_KYWD);
225 wint_t __ungetwc_xpg5(wint_t, __FILE *);

227 /*
228  * /usr/src/lib/libc/port/gen routines
229  */

231 /* _ctype.c */

233 /* _loc_data.c */

235 /* _locale.c */

237 /* _set_tab.c */
238 int __set_tab(const char *loc, int cat);

240 /* _xftw.c */
241 int __xftw(int ver, const char *path, int (*fn)(), int depth);

243 /* a64l.c */
244 long a64l(const char *);

246 /* abort.c */
247 void abort(void);

249 /* abs.c */
250 int abs(int arg);
251 long labs(long int arg);

253 /* assert.c */
254 void __assert(const char *assertion, const char *filename, int line_num);
255 void __assert_c99(const char *assertion, const char *filename, int line_num,
256 const char *funcname);

258 /* atexit.c */

```

```

259 int atexit(void(*func)());
260 void _exithandle(void);

262 /* atof.c */
263 double atof(const char *p);

265 /* atoi.c */
266 int atoi(const char *p);

268 /* atol.c */
269 long atol(const char *p);

271 /* basename.c */
272 char *basename(char *s);

274 /* bcmp.c */
275 int bcmp(const void *s1, const void *s2, size_t len);

277 /* bcopy.c */
278 void bcopy(const void *s1, void *s2, size_t len);

280 /* bsearch.c */
281 void *bsearch(const void *ky, const void *bs, size_t nel,
282              size_t width, int (*compar)());

284 /* bzero.c */
285 void bzero(void *sp, size_t len);

287 /* calloc.c */
288 void *calloc(size_t num, size_t size);

290 /* catclose.c */
291 int catclose(nl_catd catd);

293 /* catgets.c */
294 char *catgets(nl_catd catd, int set_num, int msg_num, const char *s);

296 /* catopen.c */
297 nl_catd catopen(const char *name, int mode);

299 /* cfgetispeed.c */
300 speed_t cfgetispeed(const struct termios *termios_p);

302 /* cfgetospeed.c */
303 speed_t cfgetospeed(const struct termios *termios_p);

305 /* cfree.c */
306 void cfree(void *p, size_t num, size_t size);

308 /* cfsetispeed.c */
309 int cfsetispeed(struct termios *termios_p, speed_t speed);

311 /* cfsetospeed.c */
312 int cfsetospeed(struct termios *termios_p, speed_t speed);

314 /* cftime.c */
315 int cftime(char *buf, char *format, const time_t *t);
316 int ascftime(char *buf, const char *format, const struct tm *tm);

318 /* clock.c */
319 clock_t clock(void);

321 /* closedir.c */
322 int closedir(DIR *dirp);

324 /* confstr.c */

```

```

325 size_t confstr(int name, char *buf, size_t length);

327 /* crypt.c */
328 void setkey(const char *key);
329 void encrypt(char *block, int fake);
330 char *crypt(const char *key, const char *salt);

332 /* csetlen.c */
333 int csetlen(int cset);
334 int csetcol(int cset);

336 /* ctime.c */
337 char *ctime(const time_t *t);
338 char *ctime_r(const time_t *, char *buf, int);
339 char *asctime(const struct tm *t);
340 char *asctime_r(const struct tm *, char *, int);

342 /* ctypefcns.c */
343 int isalpha(int c);
344 int isupper(int c);
345 int islower(int c);
346 int isdigit(int c);
347 int isxdigit(int c);
348 int isalnum(int c);
349 int isspace(int c);
350 int ispunct(int c);
351 int isprint(int c);
352 int isgraph(int c);
353 int iscntrl(int c);
354 int isascii(int c);
355 int _toupper(int c);
356 int _tolower(int c);
357 int toascii(int c);

359 /* daemon.c */
360 int daemon(int nochdir, int noclose);

362 /* directio.c */
363 int directio(int filedes, int advice);

365 /* dirname.c */
366 char *dirname(char *s);

368 /* div.c */
369 div_t div(int numer, int denom);
370 ldiv_t ldiv(long int numer, long int denom);

372 /* drand48.c */
373 double drand48(void);
374 double erand48(unsigned short *xsubi);
375 long krand48(unsigned short *xsubi, unsigned int m);
376 long lrand48(void);
377 long mrand48(void);
378 void srand48(long seedval);
379 unsigned short *seed48(unsigned short seed16v[3]);
380 void lcong48(unsigned short param[7]);
381 long nrand48(unsigned short *xsubi);
382 long jrand48(unsigned short *xsubi);

384 /* dup2.c */
385 int dup2(int fildes, int fildes2);

387 /* ecvt.c */
388 char *ecvt(double value, int ndigit, int *_RESTRICT_KYWD decpt,
389           int *_RESTRICT_KYWDsign);
390 char *fcvt(double value, int ndigit, int *_RESTRICT_KYWD decpt,

```

```

391     int *_RESTRICT_KYWD sign);

393 /* err.c */
394 void _errfp(FILE *, int, const char *, ...);
395 void _verrfp(FILE *, int, const char *, va_list);
396 void _errxfp(FILE *, int, const char *, ...);
397 void _verrxfp(FILE *, int, const char *, va_list);
398 void _warnfp(FILE *, const char *, ...);
399 void _vwarnfp(FILE *, const char *, va_list);
400 void _warnxfp(FILE *, const char *, ...);
401 void _vwarnxfp(FILE *, const char *, va_list);

403 /* errlst.c */

405 /* euclen.c */
406 int euccol(const unsigned char *s);
407 int euclen(const unsigned char *s);
408 int eucscol(const unsigned char *s);

410 /* execvp.c */
411 /* VARARGS1 */
412 int execlp(const char *, const char *, ...);
413 int execvp(const char *name, char *const *argv);

415 /* fattach.c */
416 int fattach(int fildes, const char *path);

418 /* fdetach.c */
419 int fdetach(const char *path);

421 /* ffs.c */
422 int ffs(int field);

424 /* fmtmsg.c */
425 int addseverity(int value, const char *string);
426 int fmtmsg(long class, const char *label, int severity, const char *text,
427             const char *action, const char *tag);

429 /* ftime.c */
430 int ftime(struct timeb *tp);

432 /* ftok.c */
433 key_t ftok(const char *path, int id);

435 /* gcvt.c */
436 char *gcvt(double number, int ndigit, char *buf);

438 /* getcwd.c */
439 char *getcwd(char *str, size_t size);

441 /* getdate.c */
442 struct tm *getdate(const char *expression);
443 #ifdef getdate_err
444 #undef getdate_err
445 #endif
446 int getdate_err;

448 /* getdate_data.c */

450 /* getdate_gd.c */

452 /* getdtblsize.c */
453 int getdtblsize(void);

455 /* getenv.c */
456 char *getenv(const char *name);

```

```

458 /* getexecname.c */
459 const char *getexecname(void);

461 /* getgrnam.c */
462 struct group *getgrnam(const char *name);
463 struct group *getgrgid(gid_t gid);
464 struct group *fgetgrent_r(FILE *, struct group *, char *, int);
465 struct group *getgrent_r(struct group *, char *, int);
466 struct group *getgrgid_r(gid_t, struct group *, char *, int);
467 struct group *getgrnam_r(const char *, struct group *, char *, int);

469 /* gethostid.c */
470 long gethostid(void);

472 /* gethz.c */
473 int gethz(void);

475 /* getisax.c */
476 uint_t getisax(uint32_t *, uint_t);

478 /* getlogin.c */
479 char *getloginx(void);
480 char *getloginx_r(char *, int);
481 #ifdef getlogin
482 #undef getlogin
483 #endif /* getlogin */
484 char *getlogin(void);
485 #ifdef getlogin_r
486 #undef getlogin_r
487 #endif /* getlogin_r */
488 char *getlogin_r(char *, int);

490 /* getmntent.c */
491 int getmntany(FILE *fd, struct mnttab *mgetp, struct mnttab *mrefp);
492 int getmntent(FILE *fd, struct mnttab *mp);

494 /* getnetgrent.c */
495 int setnetgrent(const char *grp);
496 int endnetgrent(void);
497 int getnetgrent(char **machinep, char **namep, char **domainp);

499 /* getopt.c */
500 int getopt(int argc, char *const *argv, const char *opts);

502 /* getopt_long.c */
503 int getopt_clip(int argc, char *const *argv, const char *optstring,
504                 const struct option *long_options, int *long_index);
505 int getopt_long(int argc, char *const *argv, const char *optstring,
506                 const struct option *long_options, int *long_index);
507 int getopt_long_only(int argc, char *const *argv, const char *optstring,
508                       const struct option *long_options, int *long_index);

510 /* getpagesize.c */
511 int getpagesize(void);

513 /* getpw.c */
514 int getpw(uid_t uid, char *buf);

516 /* getpwnam.c */
517 struct passwd *getpwnam(const char *name);
518 struct passwd *getpwuid(uid_t uid);
519 struct passwd *fgetpwent_r(FILE *, struct passwd *, char *, int);
520 struct passwd *getpwent_r(struct passwd *, char *, int);
521 struct passwd *getpwnam_r(const char *, struct passwd *, char *, int);
522 struct passwd *getpwuid_r(uid_t, struct passwd *, char *, int);

```

```

524 /* getrusage.c */
525 int getrusage(int who, struct rusage *rusage);

527 /* gettimeofday.c */
528 int gettimeofday(struct timeval *_RESTRICT_KYWD tp, void *_RESTRICT_KYWD);

530 /* getspent.c */
531 void setspent(void);
532 void endspent(void);
533 struct spwd *getspent(void);
534 struct spwd *getspent_r(struct spwd *, char *, int);
535 struct spwd *fgetspent(FILE *f);
536 struct spwd *fgetspent_r(FILE *, struct spwd *, char *, int);
537 struct spwd *getspnam(const char *name);
538 struct spwd *getspnam_r(const char *, struct spwd *, char *, int);
539 int putspent(const struct spwd *p, FILE *f);

541 /* getspent_r.c */
542 int str2spwd(const char *, int, void *, char *, int);

544 /* getsubopt.c */
545 int getsubopt(char **optionsp, char *const *tokens, char **valuep);

547 /* gettxt.c */
548 char *gettxt(const char *msg_id, const char *dflt_str);

550 /* getusershell.c */
551 char *getusershell(void);
552 void endusershell(void);
553 void setusershell(void);

555 /* getut.c */
556 struct utmp *getutent(void);
557 struct utmp *getutid(const struct utmp *entry);
558 struct utmp *getutline(const struct utmp *entry);
559 struct utmp *pututline(const struct utmp *entry);
560 void setutent(void);
561 void endutent(void);
562 int utmpname(const char *newfile);
563 void updwtmp(const char *file, struct utmp *ut);
564 void getutmp(const struct utmpx *utx, struct utmp *ut);
565 void getutmpx(const struct utmp *ut, struct utmpx *utx);
566 struct utmp *makeut(struct utmp *utmp);

568 /* getutx.c */
569 struct utmpx *getutxent(void);
570 struct utmpx *getutxid(const struct utmpx *entry);
571 struct utmpx *getutxline(const struct utmpx *entry);
572 struct utmpx *pututxline(const struct utmpx *entry);
573 void setutxent(void);
574 void endutxent(void);
575 int utmpxname(const char *newfile);
576 void updwtmpx(const char *file, struct utmpx *utx);
577 struct utmpx *makeutx(const struct utmpx *utmp);
578 struct utmpx *modutx(const struct utmpx *utp);

580 /* getvfsent.c */
581 int getvfsspec(FILE *fd, struct vfstab *vp, char *special);
582 int getvfsfile(FILE *fd, struct vfstab *vp, char *mountp);
583 int getvfsany(FILE *fd, struct vfstab *vgetp, struct vfstab *vrefp);
584 int getvfisent(FILE *fd, struct vfstab *vp);

586 /* getwd.c */
587 char *getwd(char *pathname);

```

```

589 /* getwidth.c */
590 void getwidth(eucwidth_t *eucstruct);

592 /* hsearch.c */
593 int hcreate(size_t size);
594 void hdestroy(void);
595 ENTRY *hsearch(ENTRY item, ACTION action);

597 /* iconv.c */
598 size_t iconv(iconv_t cd, const char **_RESTRICT_KYWD inbuf,
599             size_t *_RESTRICT_KYWD inbytesleft, char **_RESTRICT_KYWD outbuf,
600             size_t *_RESTRICT_KYWD outbytesleft);
601 int iconv_close(iconv_t cd);
602 iconv_t iconv_open(const char *tocode, const char *fromcode);

604 /* imaxabs.c */
605 intmax_t imaxabs(intmax_t j);

607 /* imaxdiv.c */
608 imaxdiv_t imaxdiv(intmax_t numer, intmax_t denom);

610 /* index.c */
611 char *index(const char *sp, int c);

613 /* initgroups.c */
614 int initgroups(const char *uname, gid_t agroup);

616 /* inetgr.c */
617 int inetgr(const char *group, const char *machine, const char *name,
618           const char *domain);

620 /* insque.c */
621 void insque(void *elem, void *pred);
622 void remque(void *elem);

624 /* isaexec.c */
625 int isaexec(const char *, char *const *, char *const *);

627 /* isastream.c */
628 int isastream(int fd);

630 /* isatty.c */
631 int isatty(int f);

633 /* killpg.c */
634 int killpg(pid_t pgrp, int sig);

636 /* l64a.c */
637 char *l64a(long lg);

639 /* lckpwwdf.c */
640 int lckpwwdf(void);
641 int ulckpwwdf(void);

643 /* lfind.c */
644 void * lfind(const void *ky, const void *bs, size_t *nel,
645             size_t width, int (*compar)());

647 /* localeconv.c */
648 struct lconv *localeconv(void);

650 /* lsearch.c */
651 void * lsearch(const void *ky, void *bs, size_t *nel,
652             size_t width, int (*compar)());

654 /* madvise.c */

```

```

655 int madvise(caddr_t addr, size_t len, int advice);

657 /* malloc.c */
658 void *malloc(size_t size);
659 void *realloc(void *old, size_t size);
660 void free(void *old);

662 /* mbstowcs.c */
663 size_t mbstowcs(wchar_t *_RESTRICT_KYWD pwcs, const char *_RESTRICT_KYWD s,
664               size_t n);

666 /* mbtowc.c */
667 int mbtowc(wchar_t *_RESTRICT_KYWD wchar, const char *_RESTRICT_KYWD s,
668           size_t n);
669 int mblen(const char *s, size_t n);

671 /* memalign.c */
672 void *memalign(size_t align, size_t nbytes);

674 /* memcpy.c */
675 void *memcpy(void *_RESTRICT_KYWDs, const void *_RESTRICT_KYWD s0, int c,
676            size_t n);

678 /* memchr.c */
679 void *memchr(const void *sptr, int cl, size_t n);

681 /* memcmp.c */
682 int memcmp(const void *s1, const void *s2, size_t n);

684 /* memcpy.c */
685 void *memcpy(void *_RESTRICT_KYWD s, const void *_RESTRICT_KYWD s0, size_t n);

687 /* memmove.c */
688 void *memmove(void *s, const void *s0, size_t n);

690 /* memset.c */
691 void *memset(void *spl, int c, size_t n);

693 /* mkdev.c */
694 dev_t __makedev(const int version, const major_t majdev,
695               const minor_t mindev);
696 major_t __major(const int version, const dev_t devnum);
697 minor_t __minor(const int version, const dev_t devnum);

699 /* mkfifo.c */
700 int mkfifo(const char *path, mode_t mode);

702 /* mktemp.c */
703 char *mktemp(char *as);

705 /* mlock.c */
706 int mlock(caddr_t addr, size_t len);

708 /* mlockall.c */
709 int mlockall(int flags);

711 /* mon.c */
712 void monitor(int (*alowpc)(), int (*ahighpc)(), WORD *buffer,
713             size_t bufsize, size_t nfunc);

715 /* msync.c */
716 int msync(caddr_t addr, size_t len, int flags);

718 /* munlock.c */
719 int munlock(caddr_t addr, size_t len);

```

```

721 /* munlockall.c */
722 int munlockall(void);

724 /* ndbm.c */
725 void dbm_setdefwrite(DBM *db);
726 int dbm_flush(DBM *db);
727 int dbm_flushpag(DBM *db);
728 DBM *dbm_open(const char *file, int flags, mode_t mode);
729 void dbm_close(DBM *db);
730 int dbm_close_status(DBM *db);
731 datum dbm_fetch(DBM *db, datum key);
732 int dbm_delete(DBM *db, datum key);
733 int dbm_store(DBM *db, datum key, datum dat, int replace);
734 datum dbm_firstkey(DBM *db);
735 datum dbm_nextkey(DBM *db);
736 datum dbm_do_nextkey(DBM *db, datum inkey);

738 /* new_list.c */

740 /* nftw.c */
741 int nftw(const char *path, int (*fn)(), int depth, int flags);

743 /* nl_langinfo.c */
744 char *nl_langinfo(nl_item item);

746 /* opendir.c */
747 DIR *opendir(const char *filename);

749 /* opt_data.c */

751 /* perror.c */
752 void perror(const char *s);

754 /* psiginfo.c */
755 void psiginfo(siginfo_t *sip, char *s);

757 /* psignal.c */
758 void psignal(int sig, const char *s);

760 /* pt.c */
761 char *ptsname(int fd);
762 int unlockpt(int fd);
763 int grantpt(int fd);

765 /* putenv.c */
766 int putenv(char *change);
767 int setenv(const char *envname, const char *envval, int overwrite);
768 int unsetenv(const char *name);

770 /* putpwent.c */
771 int putpwent(const struct passwd *p, FILE *f);

773 /* qsort.c */
774 void qsort(void *base, size_t n, size_t size, int (*compar)());

776 /* raise.c */
777 int raise(int sig);

779 /* rand.c */
780 void srand(unsigned x);
781 int rand(void);
782 int rand_r(unsigned int *);

784 /* random.c */
785 void srandom(unsigned x);
786 char *initstate(unsigned seed, char *arg_state, size_t n);

```

```

787 char *setstate(const char *arg_state);
788 long random(void);

790 /* rctlops.c */
791 int rctl_walk(int (*callback)(const char *, void *), void *walk_data);
792 hrtime_t rctlblk_get_firing_time(rctlblk_t *rblk);
793 uint_t rctlblk_get_global_action(rctlblk_t *rblk);
794 uint_t rctlblk_get_global_flags(rctlblk_t *rblk);
795 uint_t rctlblk_get_local_action(rctlblk_t *rblk, int *signalp);
796 uint_t rctlblk_get_local_flags(rctlblk_t *rblk);
797 id_t rctlblk_get_recipient_pid(rctlblk_t *rblk);
798 rctl_priv_t rctlblk_get_privilege(rctlblk_t *rblk);
799 rctl_qty_t rctlblk_get_value(rctlblk_t *rblk);
800 void rctlblk_set_local_action(rctlblk_t *rblk, uint_t action, int signal);
801 void rctlblk_set_local_flags(rctlblk_t *rblk, uint_t flags);
802 void rctlblk_set_privilege(rctlblk_t *rblk, rctl_priv_t priv);
803 void rctlblk_set_value(rctlblk_t *rblk, rctl_qty_t val);
804 size_t rctlblk_size(void);

806 /* readdir.c */
807 struct dirent *readdir(DIR *dirp);

809 /* realpath.c */
810 char *realpath(const char *_RESTRICT_KYWD raw, char *_RESTRICT_KYWD canon);

812 /* regex.c */
813 char *re_comp(const char *sp);
814 int re_exec(const char *p1);

816 /* rindex.c */
817 char *rindex(const char *sp, int c);

819 /* rename.c */
820 int remove(const char *filename);
821 int rename(const char *old, const char *new);

823 /* rewinddir.c */
824 #undef rewinddir
825 void rewinddir(DIR *dirp);

827 /* scandir.c */
828 int alphasort(const struct dirent **, const struct dirent **);
829 int scandir(const char *dirname, struct dirent *(*namelist[]),
830             int (*select)(const struct dirent *),
831             int (*dcomp)(const struct dirent **, const struct dirent **));

833 /* scrwidth.c */
834 int scrwidth(wchar_t c);

836 /* seekdir.c */
837 void seekdir(DIR *dirp, long loc);

839 /* select.c */
840 int pselect(int nfd,
841            fd_set *_RESTRICT_KYWD readfds,
842            fd_set *_RESTRICT_KYWD writefds,
843            fd_set *_RESTRICT_KYWD errorfds,
844            const struct timespec *_RESTRICT_KYWD timeout,
845            const sigset_t *_RESTRICT_KYWD sigmask);
846 int select(int nfd,
847            fd_set *_RESTRICT_KYWD readfds,
848            fd_set *_RESTRICT_KYWD writefds,
849            fd_set *_RESTRICT_KYWD errorfds,
850            struct timeval *_RESTRICT_KYWD timeout);

852 /* setlocale.c */

```

```

853 char *setlocale(int cat, const char *loc);

855 /* setpriority.c */
856 int getpriority(int which, id_t who);
857 int setpriority(int which, id_t who, int prio);

859 /* settimeofday.c */
860 int settimeofday(struct timeval *tp, void *);

862 /* sigflag.c */
863 int sigflag(int sig, int flag, int on);

865 /* siglist.c */

867 /* sigsend.c */
868 int sigsend(idtype_t idtype, id_t id, int sig);

870 /* sigsetops.c */
871 int sigfillset(sigset_t *set);
872 int sigemptyset(sigset_t *set);
873 int sigaddset(sigset_t *set, int sig);
874 int sigdelset(sigset_t *set, int sig);
875 int sigismember(const sigset_t *set, int sig);

877 /* scalls.c */
878 unsigned sleep(unsigned sleep_tm);

880 /* ssignal.c */
881 int (*ssignal(int sig, int (*fn)()) ())();
882 int gsignal(int sig);

884 /* str2id.c */

886 /* str2sig.c */
887 int str2sig(const char *s, int *sigp);
888 int sig2str(int i, char *s);

890 /* strcat.c */
891 char *strcat(char *_RESTRICT_KYWD s1, const char *_RESTRICT_KYWD s2);

893 /* strchr.c */
894 char *strchr(const char *sp, int c);

896 /* strcmp.c */
897 int strcmp(const char *s1, const char *s2);

899 /* strcpy.c */
900 char *strcpy(char *_RESTRICT_KYWD s1, const char *_RESTRICT_KYWD s2);

902 /* strcspn.c */
903 size_t strcspn(const char *string, const char *charset);

905 /* strdup.c */
906 char *strdup(const char *s1);

908 /* strerror.c */
909 char *strerror(int errnum);
910 int strerror_r(int errnum, char *strerrbuf, size_t buflen);

912 /* strftime.c */
913 size_t strftime(char *_RESTRICT_KYWD s, size_t maxsize,
914                const char *_RESTRICT_KYWD format,
915                const struct tm *_RESTRICT_KYWD tm);

917 /* strlen.c */
918 size_t strlen(const char *s);

```



```

920 /* strncat.c */
921 char *strncat(char *_RESTRICT_KYWD s1, const char *_RESTRICT_KYWD s2, size_t n);

923 /* strncmp.c */
924 int strncmp(const char *s1, const char *s2, size_t n);

926 /* strncpy.c */
927 char *strncpy(char *_RESTRICT_KYWD s1, const char *_RESTRICT_KYWD s2, size_t n);

929 /* strpbrk.c */
930 char *strpbrk(const char *string, const char *brkset);

932 /* strrchr.c */
933 char *strrchr(const char *sp, int c);

935 /* strsep.c */
936 char *strsep(char **stringp, const char *delim);

938 /* strspn.c */
939 size_t strspn(const char *string, const char *charset);

941 /* strstr.c */
942 char *strstr(const char *as1, const char *as2);

944 /* strtod.c */
945 double strtod(const char *_RESTRICT_KYWD cp, char **_RESTRICT_KYWD ptr);
946 float strtof(const char *_RESTRICT_KYWD cp, char **_RESTRICT_KYWD ptr);
947 long double strtold(const char *_RESTRICT_KYWD cp, char **_RESTRICT_KYWD ptr);

949 /* strtoumax.c */
950 intmax_t strtoumax(const char *_RESTRICT_KYWD nptr,
951 char **_RESTRICT_KYWD endptr, int base);

953 /* strtok.c */
954 char *strtok(char *_RESTRICT_KYWD string, const char *_RESTRICT_KYWD sepset);
955 char *strtok_r(char *_RESTRICT_KYWD, const char *_RESTRICT_KYWD,
956 char **_RESTRICT_KYWD);

958 /* strtol.c */
959 long strtol(const char *_RESTRICT_KYWD str, char **_RESTRICT_KYWD nptr,
960 int base);

962 /* strtoul.c */
963 unsigned long strtoul(const char *_RESTRICT_KYWD str,
964 char **_RESTRICT_KYWD nptr, int base);

966 /* strtoumax.c */
967 uintmax_t strtoumax(const char *_RESTRICT_KYWD nptr,
968 char **_RESTRICT_KYWD endptr, int base);

970 /* strxfrm.c */
971 size_t strxfrm(char *_RESTRICT_KYWD s1, const char *_RESTRICT_KYWD s2,
972 size_t n);
973 int strcoll(const char *s1, const char *s2);

975 /* swab.c */
976 void swab(const char *_RESTRICT_KYWD from, char *_RESTRICT_KYWD to, ssize_t n);

978 /* swapctl.c */
979 int swapctl(int cmd, void *arg);

981 /* sysconf.c */
982 long sysconf(int name);

984 /* syslog.c */

```

```

985 /* VARARGS2 */
986 void syslog(int pri, const char *fmt, ...);
987 void vsyslog(int pri, const char *fmt, va_list ap);
988 void openlog(const char *ident, int logstat, int logfac);
989 void closelog(void);
990 int setlogmask(int pmask);

992 /* tcdrain.c */
993 int tcdrain(int fildes);

995 /* tcflow.c */
996 int tcflow(int fildes, int action);

998 /* tcflush.c */
999 int tcflush(int fildes, int queue_selector);

1001 /* tcgetattr.c */
1002 int tcgetattr(int fildes, struct termios *termios_p);

1004 /* tcgetpgrp.c */
1005 pid_t tcgetpgrp(int fd);

1007 /* tcgetsid.c */
1008 pid_t tcgetsid(int fd);

1010 /* tcsendbreak.c */
1011 int tcsendbreak(int fildes, int duration);

1013 /* tcsetattr.c */
1014 int tcsetattr(int fildes, int optional_actions,
1015 const struct termios *termios_p);

1017 /* tcsetpgrp.c */
1018 int tcsetpgrp(int fd, pid_t pgrp);

1020 /* tell.c */
1021 long tell(int f);

1023 /* telldir.c */
1024 long telldir(DIR *dirp);

1026 /* tfind.c */
1027 void *tfind(const void *ky, void *const *rtp, int (*compar)());

1029 /* time_comm.c */
1030 struct tm *localtime(const time_t *timep);
1031 struct tm *localtime_r(const time_t *_RESTRICT_KYWD, struct tm *_RESTRICT_KYWD);
1032 struct tm *gmtime(const time_t *clock);
1033 struct tm *gmtime_r(const time_t *_RESTRICT_KYWD, struct tm *_RESTRICT_KYWD);
1034 double difftime(time_t timel, time_t time0);
1035 time_t mktime(struct tm *timep);
1036 void _ltzset(time_t tim);
1037 void tzset(void);

1039 /* time_data.c */

1041 /* time_gdata.c */

1043 /* tolower.c */
1044 int tolower(int c);

1046 /* toupper.c */
1047 int toupper(int c);

1049 /* truncate.c */
1050 int ftruncate(int fildes, off_t len);

```

```

1051 int truncate(const char *path, off_t len);

1053 /* tsearch.c */
1054 void *tsearch(const void *ky, void **rtp, int (*compar)());
1055 void *tdelete(const void *ky, void **rtp, int (*compar)());
1056 void twalk(const void *rt, void (*action)());

1058 /* ttyname.c */
1059 char *ttyname(int f);
1060 char *ttyname_dev(dev_t rdev, char *buffer, size_t buflen);
1061 char *ttyname_r(int, char *, int);

1063 /* ttyslot.c */
1064 int ttyslot(void);

1066 /* ualarm.c */
1067 unsigned ualarm(unsigned usecs, unsigned reload);

1069 /* ulimit.c */
1070 /* VARARGS1 */
1071 long ulimit(int cmd, ...);

1073 /* scalls.c */
1074 int usleep(unsigned n);

1076 /* valloc.c */
1077 void *valloc(size_t size);

1079 /* waitpid.c */
1080 pid_t wait(int *stat_loc);
1081 pid_t waitpid(pid_t pid, int *stat_loc, int options);
1082 pid_t wait3(int *status, int options, struct rusage *rp);
1083 pid_t wait4(pid_t pid, int *status, int options, struct rusage *rusage);

1085 /* wcstombs.c */
1086 size_t wcstombs(char *_RESTRICT_KYWD s, const wchar_t *_RESTRICT_KYWD pwcs,
1087               size_t n);

1089 /* wctomb.c */
1090 int wctomb(char *s, wchar_t wchar);

1092 /* wdata.c */

1094 /* wisprint.c */
1095 int wisprint(wchar_t c);

1097 /* xgetwidth.c */
1098 void _xgetwidth(void);

1100 /*
1101 * /usr/src/lib/libc/port/intl routines
1102 */

1104 /* gettext.c */
1105 char *bindtextdomain(const char *domain, const char *binding);
1106 char *dcgettext(const char *domain, const char *msg_id, const int category);
1107 char *dgettext(const char *domain, const char *msg_id);
1108 char *gettext(const char *msg_id);
1109 char *textdomain(const char *domain);

1111 /*
1112 * /usr/src/lib/libc/port/print routines
1113 */

1115 /* fprintf.c */
1116 /* VARARGS2 */

```

```

1117 int fprintf(FILE *_RESTRICT_KYWD iop, const char *_RESTRICT_KYWD format, ...);

1119 /* printf.c */
1120 /* VARARGS1 */
1121 int printf(const char *_RESTRICT_KYWD format, ...);

1123 /* snprintf.c */
1124 /* VARARGS2 */
1125 int snprintf(char *_RESTRICT_KYWD string, size_t n,
1126             const char *_RESTRICT_KYWD format, ...);

1128 /* sprintf.c */
1129 /* VARARGS2 */
1130 int sprintf(char *_RESTRICT_KYWD string,
1131            const char *_RESTRICT_KYWD format, ...);

1133 /* vfprintf.c */
1134 /* VARARGS2 */
1135 int vfprintf(FILE *_RESTRICT_KYWD iop, const char *_RESTRICT_KYWD format,
1136             va_list);

1138 /* vprintf.c */
1139 /* VARARGS1 */
1140 int vprintf(const char *_RESTRICT_KYWD format, va_list);

1142 /* vsnprintf.c */
1143 /* VARARGS2 */
1144 int vsnprintf(char *_RESTRICT_KYWD string, size_t n,
1145             const char *_RESTRICT_KYWD format, va_list);

1147 /* vsprintf.c */
1148 /* VARARGS2 */
1149 int vsprintf(char *_RESTRICT_KYWD string, const char *_RESTRICT_KYWD format,
1150             va_list);

1152 /*
1153 * /usr/src/lib/libc/port/regex routines
1154 */

1156 /* glob.c */
1157 extern int glob(const char *restrict pattern, int flags,
1158               int (*errfunc)(const char *epath, int eerrno), glob_t *restrict pglob);
1159 extern void globfree(glob_t *pglob);

1161 /* regex.c */
1162 char *regex(const char *regexp, const char *stringp, ...);
1163 #ifdef __locl
1164 #undef __locl
1165 #endif
1166 char *__locl;

1168 /* regcmp.c */
1169 char *regcmp(const char *regexp, ...);
1170 #ifdef __i_size
1171 #undef __i_size
1172 #endif
1173 int __i_size;

1175 /*
1176 * /usr/src/lib/libc/port/stdio routines
1177 */

1179 /* _filbuf.c */
1180 int _filbuf(FILE *iop);

1182 /* _flsbuf.c */

```

```

1183 int _flsbuf(int ch, FILE *iop);

1185 /* _wrtchk.c */
1186 int _wrtchk(FILE *iop);

1188 /* clearerr.c */
1189 void clearerr(FILE *iop);

1191 /* ctermid.c */
1192 char *ctermid(char *s);
1193 char *ctermid_r(char *s);

1195 /* cuserid.c */
1196 char *cuserid(char *s);

1198 /* data.c */

1200 /* doscan.c */
1201 int _doscan(FILE *iop, const char *fmt, va_list va_alist);

1203 /* fdopen.c */
1204 FILE *fdopen(int fd, const char *type);

1206 /* feof.c */
1207 int feof(FILE *iop);

1209 /* ferror.c */
1210 int ferror(FILE *iop);

1212 /* fgetc.c */
1213 int fgetc(FILE *iop);

1215 /* fgets.c */
1216 char *fgets(char *_RESTRICT_KYWD buf, int size, FILE *_RESTRICT_KYWD iop);

1218 /* fileno.c */
1219 int _fileno(FILE *iop);

1221 /* flush.c */
1222 void _cleanup(void);
1223 FILE *_findiop(void);
1224 typedef unsigned char Uchar;
1225 void _setbufend(FILE *iop, Uchar *end);
1226 Uchar *_realbufend(FILE *iop);
1227 void _bufsync(FILE *iop, Uchar *bufend);
1228 int _xflsbuf(FILE *iop);
1229 int fflush(FILE *iop);
1230 int fclose(FILE *iop);

1232 /* fopen.c */
1233 FILE *fopen(const char *_RESTRICT_KYWD name, const char *_RESTRICT_KYWD type);
1234 FILE *freopen(const char *_RESTRICT_KYWD name, const char *_RESTRICT_KYWD type,
1235 FILE *_RESTRICT_KYWD iop);

1237 /* fpos.c */
1238 int fgetpos(FILE *_RESTRICT_KYWD stream, fpos_t *_RESTRICT_KYWD pos);
1239 int fsetpos(FILE *stream, const fpos_t *pos);

1241 /* fputc.c */
1242 int fputc(int ch, FILE *iop);

1244 /* fputs.c */
1245 int fputs(const char *_RESTRICT_KYWD ptr, FILE *_RESTRICT_KYWD iop);

1247 /* fread.c */
1248 size_t fread(void *_RESTRICT_KYWD ptr, size_t size, size_t count,

```

```

1249 FILE *_RESTRICT_KYWD iop);

1251 /* fseek.c */
1252 int fseek(FILE *iop, long offset, int ptrname);

1254 /* ftell.c */
1255 long ftell(FILE *iop);

1257 /* fwrite.c */
1258 size_t fwrite(const void *_RESTRICT_KYWD ptr1, size_t size, size_t count,
1259 FILE *_RESTRICT_KYWD iop);

1261 /*getc.c */
1262 int getc(FILE *iop);

1264 /* getchar.c */
1265 int getchar(void);

1267 /* getpass.c */
1268 char *getpass(const char *prompt);

1270 /* getpass.c */
1271 char *getpassphrase(const char *prompt);

1273 /* gets.c */
1274 char *gets(char *buf);

1276 /* getw.c */
1277 int getw(FILE *stream);

1279 /* popen.c */
1280 FILE *popen(const char *cmd, const char *mode);
1281 int pclose(FILE *ptr);

1283 /* putc.c */
1284 int putc(int ch, FILE *iop);

1286 /* putchar.c */
1287 int putchar(int ch);

1289 /* puts.c */
1290 int puts(const char *ptr);

1292 /* putw.c */
1293 int putw(int w, FILE *stream);

1295 /* rewind.c */
1296 void rewind(FILE *iop);

1298 /* scanf.c */
1299 /* VARARGS1 */
1300 int scanf(const char *_RESTRICT_KYWD fmt, ...);

1302 /* VARARGS2 */
1303 int fscanf(FILE *_RESTRICT_KYWD iop, const char *_RESTRICT_KYWD fmt, ...);

1305 /* VARARGS2 */
1306 int sscanf(const char *_RESTRICT_KYWD str, const char *_RESTRICT_KYWD fmt, ...);

1308 /* setbuf.c */
1309 void setbuf(FILE *_RESTRICT_KYWD iop, char *_RESTRICT_KYWD abuf);

1311 /* setvbuf.c */
1312 int setvbuf(FILE *_RESTRICT_KYWD iop, char *_RESTRICT_KYWD abuf, int type,
1313 size_t size);

```

```

1315 /* system.c */
1316 int system(const char *s);

1318 /* tmpnam.c */
1319 char *tmpnam(const char *dir, const char *pfx);

1321 /* tmpfile.c */
1322 FILE *tmpfile(void);

1324 /* tmpnam.c */
1325 char *tmpnam(char *s);
1326 char *tmpnam_r(char *);

1328 /* ungetc.c */
1329 int ungetc(int c, FILE *iop);

1331 /*
1332  * /usr/src/lib/libc/port/sys routines
1333  */

1335 /* exactsys.c */
1336 size_t getacct(idtype_t idtype, id_t id, void *buf, size_t bufsize);
1337 int putacct(idtype_t idtype, id_t id, void *buf, size_t bufsize, int flags);
1338 int wracct(idtype_t idtype, id_t id, int flags);

1340 /* execl.c */
1341 /* VARARGS1 */
1342 int execl(const char *name, const char *, ...);

1344 /* execl.c */
1345 int execl(const char *, const char *file, ...);

1347 /* execv.c */
1348 int execv(const char *file, char *const *argv);

1350 /* lockf.c */
1351 int lockf(int fildes, int function, off_t size);

1353 /* meminfosys.c */
1354 int meminfo(const uint64_t *inaddr, int addr_count, const uint_t *info_req,
1355             int info_count, uint64_t *outdata, uint_t *validity);

1357 /* msgsys.c */
1358 int msgget(key_t key, int msgflg);
1359 int msgctl(int msqid, int cmd, struct msqid_ds *buf);
1360 ssize_t msgrcv(int msqid, void *msgp, size_t msgsz, long msgtyp, int msgflg);
1361 int msgsnd(int msqid, const void *msgp, size_t msgsz, int msgflg);

1363 /* nfssys.c */
1364 /*
1365 int exportfs(char *dir, struct export *ep);
1366 int nfs_getfh(char *path, fh_t *fhp);
1367 int nfssvc(int fd);
1368 */

1370 /* psetsys.c */
1371 int pset_create(psetid_t *npset);
1372 int pset_destroy(psetid_t pset);
1373 int pset_assign(psetid_t pset, processorid_t cpu, psetid_t *opset);
1374 int pset_assign_forced(psetid_t pset, processorid_t cpu, psetid_t *opset);
1375 int pset_info(psetid_t pset, int *type, u_int *numcpus, processorid_t *cpulist);
1376 int pset_bind(psetid_t pset, idtype_t idtype, id_t id, psetid_t *opset);
1377 int pset_bind_lwp(psetid_t pset, id_t id, pid_t, psetid_t *opset);
1378

1380 /* rctlsys.c */

```

```

1381 int getrctl(const char *name, rctlblk_t *old_rblk, rctlblk_t *new_rblk,
1382             int flags);
1383 int setrctl(const char *name, rctlblk_t *old_rblk, rctlblk_t *new_rblk,
1384             int flags);
1385 /* (private functions) */
1386 int setprojctl(const char *name, rctlblk_t *new_rblk, size_t size, int flags);
1387 int rctlctl(const char *, rctlblk_t *, int);
1388 size_t rctlolist(char *, size_t);

1391 /* semsys.c */
1392 int semctl(int semid, int semnum, int cmd, ...);
1393 int semget(key_t key, int nsems, int semflg);
1394 int semop(int semid, struct sembuf *sops, size_t nsops);

1396 /* shmsys.c */
1397 void *shmat(int shmid, const void *shmaddr, int shmflg);
1398 int shmctl(int shmid, int cmd, struct shmids *buf);
1399 #if defined(_XOPEN_SOURCE) && (_XOPEN_VERSION - 0 == 4)
1400 int shmctl(const void *);
1401 #else
1402 int shmctl(char *);
1403 #endif /* defined(_XOPEN_SOURCE) && (_XOPEN_VERSION - 0 == 4) */
1404 int shmget(key_t key, size_t size, int shmflg);

1406 /* tasksys.c */
1407 taskid_t settaskid(projid_t project, uint_t flags);
1408 taskid_t gettaskid(void);
1409 projid_t getprojid(void);

1411 /*
1412  * /usr/src/lib/libc/port/widec routines
1413  */

1415 /* fgetws.c */
1416 wchar_t *fgetws(wchar_t *_RESTRICT_KYWD ptr, int size,
1417                 FILE *_RESTRICT_KYWD iop);

1419 /* fputwc.c */
1420 wint_t fputwc(wint_t wc, FILE *iop);
1421 wint_t putwc(wint_t wc, FILE *iop);

1423 /* fputws.c */
1424 int fputws(const wchar_t *_RESTRICT_KYWD ptr, FILE *_RESTRICT_KYWD iop);

1426 /* getwchar.c */
1427 wint_t getwchar(void);

1429 /* getwidth.c */
1430 void getwidth(eucwidth_t *eucstruct);

1432 /* getws.c */
1433 wchar_t *getws(wchar_t *ptr);

1435 /* iswctype.c */
1436 int iswctype(wint_t wc, wctype_t charclass);
1437 int iswalph(wint_t c);
1438 int iswupper(wint_t c);
1439 int iswlower(wint_t c);
1440 int iswdigit(wint_t c);
1441 int iswxdigit(wint_t c);
1442 int iswalnum(wint_t c);
1443 int iswspace(wint_t c);
1444 int iswpunct(wint_t c);
1445 int iswprint(wint_t c);
1446 int iswgraph(wint_t c);

```

```

1447 int iswcntrl(wint_t c);
1448 int isphonogram(wint_t c);
1449 int isideogram(wint_t c);
1450 int isenglish(wint_t c);
1451 int isnumber(wint_t c);
1452 int isspecial(wint_t c);

1454 /* libwcollate.c */

1456 /* putwchar.c */
1457 wint_t putwchar(wint_t c);

1459 /* putws.c */
1460 int putws(const wchar_t *ptr);

1462 /* scrwidth.c */

1464 /* strtows.c */
1465 wchar_t *strtows(wchar_t *s1, char *s2);
1466 char *wstostr(char *s1, wchar_t *s2);

1468 /* trwctype.c */
1469 wint_t towupper(wint_t c);
1470 wint_t towlower(wint_t c);

1472 /* ungetwc.c */
1473 wint_t ungetwc(wint_t wc, FILE *iop);

1475 /* wcollate.c */
1476 size_t wcsxfrm(wchar_t *_RESTRICT_KYWD s1, const wchar_t *_RESTRICT_KYWD s2,
1477               size_t n);
1478 int wscoll(const wchar_t *s1, const wchar_t *s2);

1480 /* wcsftime.c */
1481 #if !defined(__amd64) /* XX64 - fix me */
1482 size_t wcsftime(wchar_t *wcs, size_t maxsize,
1483               const char *format, const struct tm *timeptr);
1484 #endif /* __amd64 */

1486 /* wcstring.c */
1487 wint_t fgetwc(FILE *iop);
1488 wint_t getwc(FILE *iop);
1489 int wwidth(wchar_t wc);
1490 int wcswidth(const wchar_t *pwcs, size_t n);

1492 /* wcswcs.c */
1493 wchar_t *wcswcs(const wchar_t *ws1, const wchar_t *ws2);

1495 /* wcsxfrm.c - empty file! */

1497 /* wcsxfrm.xpg4.c */

1499 /* wisprint.c */
1500 int wisprint(wchar_t c);

1502 /* wscasecmp.c */
1503 int wscasecmp(const wchar_t *s1, const wchar_t *s2);

1505 /* wscat.c */
1506 wchar_t *wscat(wchar_t *_RESTRICT_KYWD s1, const wchar_t *_RESTRICT_KYWD s2);
1507 wchar_t *wscat(wchar_t *s1, const wchar_t *s2);

1509 /* wschr.c */
1510 wchar_t *wschr(const wchar_t *sp, wchar_t c);
1511 wchar_t *wschr(const wchar_t *sp, wchar_t c);

```

```

1513 /* wscmp.c */
1514 int wscmp(const wchar_t *s1, const wchar_t *s2);
1515 int wscmp(const wchar_t *s1, const wchar_t *s2);

1517 /* wscol.c */
1518 int wscol(const wchar_t *s1);

1520 /* wscopy.c */
1521 wchar_t *wscopy(wchar_t *_RESTRICT_KYWD s1, const wchar_t *_RESTRICT_KYWD s2);
1522 wchar_t *wscopy(wchar_t *s1, const wchar_t *s2);

1524 /* wscspn.c */
1525 size_t wscspn(const wchar_t *string, const wchar_t *charset);
1526 size_t wscspn(const wchar_t *string, const wchar_t *charset);

1528 /* wsdup.c */
1529 wchar_t *wsdup(const wchar_t *s1);

1531 /* wslen.c */
1532 size_t wslen(const wchar_t *s);
1533 size_t wslen(const wchar_t *s);

1535 /* wsncasecmp.c */
1536 int wsncasecmp(const wchar_t *s1, const wchar_t *s2, size_t n);

1538 /* wsnecat.c */
1539 wchar_t *wsnecat(wchar_t *_RESTRICT_KYWD s1, const wchar_t *_RESTRICT_KYWD s2,
1540                size_t n);
1541 wchar_t *wsnecat(wchar_t *s1, const wchar_t *s2, size_t n);

1543 /* wsncmp.c */
1544 int wsncmp(const wchar_t *s1, const wchar_t *s2, size_t n);
1545 int wsncmp(const wchar_t *s1, const wchar_t *s2, size_t n);

1547 /* wsncpy.c */
1548 wchar_t *wsnncpy(wchar_t *_RESTRICT_KYWD s1, const wchar_t *_RESTRICT_KYWD s2,
1549                size_t n);
1550 wchar_t *wsnncpy(wchar_t *s1, const wchar_t *s2, size_t n);

1552 /* wspbrc.c */
1553 wchar_t *wspbrc(const wchar_t *string, const wchar_t *brkset);
1554 wchar_t *wspbrc(const wchar_t *string, const wchar_t *brkset);

1556 /* wsprintf.c */
1557 int wsprintf(wchar_t *wstring, const char *format, ...);

1559 /* wsrchr.c */
1560 wchar_t *wsrchr(const wchar_t *sp, wchar_t c);
1561 wchar_t *wsrchr(const wchar_t *sp, wchar_t c);

1563 /* wsscanf.c */
1564 int wsscanf(wchar_t *s, const char *format, ...);

1566 /* wssize.c */

1568 /* wsspncpy.c */
1569 size_t wsspncpy(const wchar_t *string, const wchar_t *charset);
1570 size_t wsspncpy(const wchar_t *string, const wchar_t *charset);

1572 /* wstod.c */
1573 double wstod(const wchar_t *_RESTRICT_KYWD cp, wchar_t **_RESTRICT_KYWD ptr);
1574 float wstof(const wchar_t *_RESTRICT_KYWD cp, wchar_t **_RESTRICT_KYWD ptr);
1575 long double wstold(const wchar_t *_RESTRICT_KYWD cp,
1576                  wchar_t **_RESTRICT_KYWD ptr);
1577 double wstod(const wchar_t *cp, wchar_t **ptr);

```

```

1579 /* wstok.c */
1580 #if !defined(__amd64)          /* XX64 - fix me */
1581 wchar_t *wstok(wchar_t *string, const wchar_t *sepset);
1582 wchar_t *wstok(wchar_t *string, const wchar_t *sepset);
1583 #endif /* __amd64 */

1585 /* wcstol.c */
1586 long wcstol(const wchar_t *_RESTRICT_KYWD str, wchar_t **_RESTRICT_KYWD ptr,
1587             int base);
1588 long long wcstoll(const wchar_t *_RESTRICT_KYWD str,
1589                  wchar_t **_RESTRICT_KYWD ptr, int base);

1591 /* wcstoul.c */
1592 unsigned long wcstoul(const wchar_t *_RESTRICT_KYWD str,
1593                       wchar_t **_RESTRICT_KYWD ptr, int base);
1594 unsigned long long wcstoull(const wchar_t *_RESTRICT_KYWD str,
1595                              wchar_t **_RESTRICT_KYWD ptr, int base);

1597 /* wcstoimax.c */
1598 intmax_t wcstoimax(const wchar_t *_RESTRICT_KYWD nptr,
1599                   wchar_t **_RESTRICT_KYWD endptr, int base);
1600 uintmax_t wcstoumax(const wchar_t *_RESTRICT_KYWD nptr,
1601                     wchar_t **_RESTRICT_KYWD endptr, int base);

1603 /* wstol.c */
1604 long wstol(const wchar_t *str, wchar_t **ptr, int base);

1606 /* wstoll.c */
1607 long long wstoll(const wchar_t *str, wchar_t **ptr, int base);
1608 long long watoll(const wchar_t *p);

1610 /* wsxfrm.c */
1611 size_t wsxfrm(wchar_t *s1, const wchar_t *s2, size_t n);
1612 int wscoll(const wchar_t *s1, const wchar_t *s2);

1614 /*
1615  * /usr/src/lib/libc/port/gen/event_port.c
1616  */
1617 int port_dispatch(int port, int flags, int source, int events, uintptr_t object,
1618                  void *user);

1620 /*
1621  * /usr/src/lib/libc/$MACH/gen routines
1622  */

1624 /* alloca.s */

1626 void *__builtin_alloca(size_t);

1628 /*
1629  * modctl(int arg, ...) and utssys(...) are not available from a header
1630  * file, but our utilities which make use of it should be able to be
1631  * lint clean.
1632  */
1633 int modctl(int arg, ...);
1634 int utssys(void *buf, int arg, int type, void *outbp);

1637 typedef float single;
1638 typedef unsigned extended[3];
1639 typedef long double quadruple;
1640 typedef unsigned fp_exception_field_type;

1642 typedef char decimal_string[512];

1644 enum fp_class_type {

```

```

1645         fp_zero = 0,
1646         fp_subnormal = 1,
1647         fp_normal = 2,
1648         fp_infinity = 3,
1649         fp_quiet = 4,
1650         fp_signaling = 5
1651 };
unchanged_portion_omitted

```

```

*****
54423 Sat Jun 1 10:02:51 2013
new/usr/src/lib/libc/port/mapfile-vers
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
23 #
24 # Copyright 2010 Nexenta Systems, Inc. All rights reserved.
25 # Use is subject to license terms.
26 #
27 # Copyright (c) 2012 by Delphix. All rights reserved.
28 #
29 # Copyright (c) 2013 Gary Mills
30 #

32 #
33 # MAPFILE HEADER START
34 #
35 # WARNING: STOP NOW. DO NOT MODIFY THIS FILE.
36 # Object versioning must comply with the rules detailed in
37 #
38 #     usr/src/lib/README.mapfiles
39 #
40 # You should not be making modifications here until you've read the most current
41 # copy of that file. If you need help, contact a gatekeeper for guidance.
42 #
43 # MAPFILE HEADER END
44 #

46 $mapfile_version 2

48 #
49 # All function names added to this or any other libc mapfile
50 # must be placed under the 'protected:' designation.
51 # The 'global:' designation is used *only* for data
52 # items and for the members of the malloc() family.
53 #

55 #
56 # README README README README README README: how to update this file
57 # 1) each version of Solaris/OpenSolaris gets a version number.
58 # (Actually since Solaris is actually a series of OpenSolaris releases
59 # we'll just use OpenSolaris for this exercise.)
60 #     OpenSolaris 2008.11 gets 1.23

```

```

61 #     OpenSolaris 2009.04 gets 1.24
62 #     etc.
63 # 2) each project integration uses a unique version number.
64 #     PSARC/2008/123 gets 1.24.1
65 #     PSARC/2008/456 gets 1.24.2
66 #     etc.
67 #

70 # Mnemonic conditional input identifiers:
71 #
72 # - amd64, i386, sparc32, sparcv9: Correspond to ISA subdirectories used to
73 #   hold per-platform code. Note however that we use 'sparc32' instead of
74 #   'sparc'. Since '_sparc' is predefined to apply to, all sparc platforms,
75 #   naming the 32-bit version 'sparc' would be too likely to cause errors.
76 #
77 # - lf64: Defined on platforms that offer the 32-bit largefile APIs
78 #
79 $if _ELF32
80 $add lf64
81 $endif
82 $if _sparc && _ELF32
83 $add sparc32
84 $endif
85 $if _sparc && _ELF64
86 $add sparcv9
87 $endif
88 $if _x86 && _ELF32
89 $add i386
90 $endif
91 $if _x86 && _ELF64
92 $add amd64
93 $endif

95 SYMBOL_VERSION ILLUMOS_0.3 { # Illumos additions
96     protected:
97         assfail3;
98 } ILLUMOS_0.2;
_____unchanged_portion_omitted_____

2540 # There should never be more than one SUNWprivate version.
2541 # Don't add any more. Add new private symbols to SUNWprivate_1.1

2543 SYMBOL_VERSION SUNWprivate_1.1 {
2544     global:
2545         __Argv                { FLAGS = NODIRECT };
2546         cfree                  { FLAGS = NODIRECT };
2547         __cswidth;
2548         __ctype_mask;
2549         __environ_lock        { FLAGS = NODIRECT };
2550         __inf_read;
2551         __inf_written;
2552         __i_size;
2553         __isnanf                { TYPE = FUNCTION; FILTER = libm.so.2 };
2554         __iswrunes;
2555         __libc_threaded;
2556         __lib_version          { FLAGS = NODIRECT };
2557         __logb                  { TYPE = FUNCTION; FILTER = libm.so.2 };
2558         __lone                  { FLAGS = NODYNSORT };
2559         __lten                  { FLAGS = NODYNSORT };
2560         __lzero                 { FLAGS = NODYNSORT };
2561         __malloc_lock;
2562         __memcmp;
2563         __memcpy                { FLAGS = NODYNSORT };

```

```

2564     __memmove;
2565     __memset;
2566     __modff           { TYPE = FUNCTION; FILTER = libm.so.2 };
2567     __nan_read;
2568     __nan_written;
2569     __nextwctype;
2570     __nis_debug_bind;
2571     __nis_debug_calls;
2572     __nis_debug_file;
2573     __nis_debug_rpc;
2574     __nis_prefsrv;
2575     __nis_preftype;
2576     __nis_server;
2577     __nss_default_finders;
2578     __progname       { FLAGS = NODIRECT };
2579     __smbuf;
2580     __sp;
2581     __strdupa_str    { FLAGS = NODIRECT };
2582     __strdupa_len    { FLAGS = NODIRECT };
2583     __tdb_bootstrap;
2584     __threaded;
2585     thr_probe_getfunc_addr;
2586     __trans_lower;
2587     __trans_upper;
2588     __uberdata;
2589     __xpg6           { FLAGS = NODIRECT };

2591 $if _ELF32
2592     __dladdr         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2593     __dladdr1        { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2594     __dlclose        { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2595     __dldump         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2596     __dlerror        { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2597     __dlinfo         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2598     __dlmopen        { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2599     __dlopen         { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2600     __dlsym          { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2601     __ld_libc        { TYPE = FUNCTION; FILTER = /usr/lib/ld.so.1 };
2602     __sys_errlist;
2603     __sys_errs;
2604     __sys_index;
2605     __sys_nerr       { FLAGS = NODYNSORT };
2606     __sys_num_err;
2607 $elif sparcv9
2608     __dladdr         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2609     __dladdr1        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2610     __dlclose        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2611     __dldump         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2612     __dlerror        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2613     __dlinfo         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2614     __dlmopen        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2615     __dlopen         { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2616     __dlsym          { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2617     __ld_libc        { TYPE = FUNCTION; FILTER = /usr/lib/sparcv9/ld.so.1 };
2618 $elif amd64
2619     __dladdr         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2620     __dladdr1        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2621     __dlamd64getunwind { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2622     __dlclose        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2623     __dldump         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2624     __dlerror        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2625     __dlinfo         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2626     __dlmopen        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2627     __dlopen         { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2628     __dlsym          { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };
2629     __ld_libc        { TYPE = FUNCTION; FILTER = /usr/lib/amd64/ld.so.1 };

```

```

2630 $else
2631 $error unknown platform
2632 $endif

2634 $if _sparc
2635     __lyday_to_month;
2636     __mon_lengths;
2637     __yday_to_month;
2638 $endif
2639 $if i386
2640     __sse_hw;
2641 $endif

2643     protected:
2644     acctctl;
2645     allocids;
2646     __assert_c99;
2647     __assert_c99;
2648     __assfail;
2649     attr_count;
2650     attr_to_data_type;
2651     attr_to_name;
2652     attr_to_option;
2653     attr_to_xattr_view;
2654     __autofssys;
2655     __bufsync;
2656     __cladm;
2657     __class_quadruple;
2658     core_get_default_content;
2659     core_get_default_path;
2660     core_get_global_content;
2661     core_get_global_path;
2662     core_get_options;
2663     core_get_process_content;
2664     core_get_process_path;
2665     core_set_default_content;
2666     core_set_default_path;
2667     core_set_global_content;
2668     core_set_global_path;
2669     core_set_options;
2670     core_set_process_content;
2671     core_set_process_path;
2672     dbm_close_status;
2673     dbm_do_nextkey;
2674     dbm_setdefwrite;
2675     __D_cplx_div;
2676     __D_cplx_div_ix;
2677     __D_cplx_div_rx;
2678     __D_cplx_mul;
2679     defclose_r;
2680     defcntl;
2681     defcntl_r;
2682     defopen;
2683     defopen_r;
2684     defread;
2685     defread_r;
2686     __delete;
2687     __dgettext;
2688     __doprnt;
2689     __doscan;
2690     __errfp;
2691     __errxfp;
2692     exportfs;
2693     __F_cplx_div;
2694     __F_cplx_div_ix;
2695     __F_cplx_div_rx;

```



```

2696     _F_cplx_mul;
2697     __fgetwc_xpg5;
2698     __fgetws_xpg5;
2699     _findbuf;
2700     _findiop;
2701     __fini_daemon_priv;
2702     _finite;
2703     _fork1                { FLAGS = NODYNSORT };
2704     _forkall             { FLAGS = NODYNSORT };
2705     _fpclass;
2706     _fpgetmask;
2707     _fpgetround;
2708     _fpgetsticky;
2709     _fprintf;
2710     _fpsetmask;
2711     _fpsetround;
2712     _fpsetsticky;
2713     __fputwc_xpg5;
2714     __fputws_xpg5;
2715     _ftw;
2716     _gcvt;
2717     _getarg;
2718     __getcontext;
2719     _getdents;
2720     _get_exit_frame_monitor;
2721     _getfp;
2722     _getgroupsbymember;
2723     _getlogin_r;
2724     getloginx;
2725     getloginx_r;
2726     _getsp;
2727     _gettsp;
2728     getvmusage;
2729     __getwchar_xpg5;
2730     __getwc_xpg5;
2731     gtty;
2732     __idmap_flush_kcache;
2733     __idmap_reg;
2734     __idmap_unreg;
2735     __init_daemon_priv;
2736     __init_suid_priv;
2737     _insert;
2738     inst_sync;
2739     _iswctype;
2740     klpd_create;
2741     klpd_getpath;
2742     klpd_getport;
2743     klpd_getucred;
2744     klpd_register;
2745     klpd_register_id;
2746     klpd_unregister;
2747     klpd_unregister_id;
2748     __lgrp_home_fast    { FLAGS = NODYNSORT };
2749     __lgrpsys;
2750     __lltostr;
2751     _lock_clear;
2752     _lock_try;
2753     _ltzset;
2754     lwp_self;
2755     makeut;
2756     makeutx;
2757     __mbftowc;
2758     mofiller;
2759     mntopt;
2760     modctl;
2761     modutx;

```

```

2762     msgctl64;
2763     __multi_innetgr;
2764     __mutex_destroy     { FLAGS = NODYNSORT };
2765     mutex_held;
2766     __mutex_init       { FLAGS = NODYNSORT };
2767     __mutex_unlock     { FLAGS = NODYNSORT };
2768     name_to_attr;
2769     nfs_getfh;
2770     nfssvc;
2771     _nfssys;
2772     __nis_get_environment;
2773     __nss_db_state_destr;
2774     nss_default_key2str;
2775     nss_delete;
2776     nss_endent;
2777     nss_getent;
2778     __nss_initf_group;
2779     __nss_initf_netgroup;
2780     __nss_initf_passwd;
2781     __nss_initf_shadow;
2782     nss_packed_arg_init;
2783     nss_packed_context_init;
2784     nss_packed_getkey;
2785     nss_packed_set_status;
2786     nss_search;
2787     nss_setent;
2788     __nss_XbyY_fgets;
2789     __nsw_extended_action_v1;
2790     __nsw_freeconfig_v1;
2791     __nsw_getconfig_v1;
2792     _nthreads;
2793     __openattdir;
2794     option_to_attr;
2795     posix_getloginx_r;
2796     __priv_bracket;
2797     __priv_relinquish;
2798     pset_assign_forced;
2799     pset_bind_lwp;
2800     _psignal;
2801     _pthread_setcleanupinit;
2802     __putwchar_xpg5;
2803     __putwc_xpg5;
2804     rctlctl;
2805     rctl1list;
2806     _realbufend;
2807     _resume;
2808     _resume_ret;
2809     _rpcsys;
2810     _sbrk_grow_aligned;
2811     scrwidth;
2812     semctl64;
2813     _semctl64;
2814     set_setcontext_enforcement;
2815     _setbufend;
2816     __set_errno;
2817     setprojctl;
2818     _setregid;
2819     _setreuid;
2820     setsigacthandler;
2821     shmctl64;
2822     __shmctl64;
2823     sigflag;
2824     _signal;
2825     _sigoff;
2826     _sigon;
2827     __so_accept;

```

```

2828     _so_bind;
2829     _sockconfig;
2830     _so_connect;
2831     _so_getpeername;
2832     _so_getsockname;
2833     _so_getsockopt;
2834     _so_listen;
2835     _so_recv;
2836     _so_recvfrom;
2837     _so_recvmsg;
2838     _so_send;
2839     _so_sendmsg;
2840     _so_sendto;
2841     _so_setsockopt;
2842     _so_shutdown;
2843     _so_socket;
2844     _so_socketpair;
2845     str2group;
2846     str2passwd;
2847     str2spwd;
2848     __strptime_dontzero;
2849     stty;
2850     syscall;
2851     _sysconfig;
2852     __systemcall;
2853     thr_continue_allmutators;
2854     _thr_continue_allmutators;
2855     thr_continue_mutator;
2856     _thr_continue_mutator;
2857     thr_getstate;
2858     _thr_getstate;
2859     thr_mutators_barrier;
2860     _thr_mutators_barrier;
2861     thr_probe_setup;
2862     _thr_schedctl;
2863     thr_setmutator;
2864     _thr_setmutator;
2865     thr_setstate;
2866     _thr_setstate;
2867     thr_sighndlrinfo;
2868     _thr_sighndlrinfo;
2869     _thr_slot_offset;
2870     thr_suspend_allmutators;
2871     _thr_suspend_allmutators;
2872     thr_suspend_mutator;
2873     _thr_suspend_mutator;
2874     thr_wait_mutator;
2875     _thr_wait_mutator;
2876     __tls_get_addr;
2877     tpool_create;
2878     tpool_dispatch;
2879     tpool_destroy;
2880     tpool_wait;
2881     tpool_suspend;
2882     tpool_suspended;
2883     tpool_resume;
2884     tpool_member;
2885     _ttyname_dev;
2886     _ucred_alloc;
2887     ucred_getamask;
2888     _ucred_getamask;
2889     ucred_getasid;
2890     _ucred_getasid;
2891     ucred_getatid;
2892     _ucred_getatid;
2893     ucred_getauid;

```

```

2894     _ucred_getauid;
2895     _ulltostr;
2896     _uncached_getgrgid_r;
2897     _uncached_getgrnam_r;
2898     _uncached_getpwnam_r;
2899     _uncached_getpwuid_r;
2900     __ungetwc_xpg5;
2901     _unordered;
2902     utssys;
2903     _verrpf;
2904     _verrxfp;
2905     _vwarnfp;
2906     _vwarnxfp;
2907     _warnfp;
2908     _warnxfp;
2909     __wcsftime_xpg5;
2910     __wcstok_xpg5;
2911     wdbindf;
2912     wdchkind;
2913     wddelim;
2914     _wrtchk;
2915     _xflsbuf;
2916     _xgetwidth;
2917     zone_add_datalink;
2918     zone_boot;
2919     zone_check_datalink;
2920     zone_create;
2921     zone_destroy;
2922     zone_enter;
2923     zone_getattr;
2924     zone_get_id;
2925     zone_list;
2926     zone_list_datalink;
2927     zonept;
2928     zone_remove_datalink;
2929     zone_setattr;
2930     zone_shutdown;
2931     zone_version;

2933 $if _ELF32
2934     __divdi3;
2935     _file_set;
2936     _fprintf_c89;
2937     _fscanf_c89;
2938     _fwprintf_c89;
2939     _fwscanf_c89;
2940     _imaxabs_c89;
2941     _imaxdiv_c89;
2942     __moddi3;
2943     _printf_c89;
2944     _scanf_c89;
2945     _snprintf_c89;
2946     _sprintf_c89;
2947     _sscanf_c89;
2948     _strtoimax_c89;
2949     _strtoumax_c89;
2950     _swprintf_c89;
2951     _swscanf_c89;
2952     __udivdi3;
2953     __umoddi3;
2954     _vfprintf_c89;
2955     _vfscanf_c89;
2956     _vfwprintf_c89;
2957     _vfwscanf_c89;
2958     _vprintf_c89;
2959     _vscanf_c89;

```

```

2960     _vsprintf_c89;
2961     _vsprintf_c89;
2962     _vsscanf_c89;
2963     _vswprintf_c89;
2964     _vswscanf_c89;
2965     _vwprintf_c89;
2966     _vscanf_c89;
2967     _wcstoimax_c89;
2968     _wcstoumax_c89;
2969     _wprintf_c89;
2970     _wscanf_c89;
2971 $endif

2973 $if _sparc
2974     _cerror;
2975     install_utrap;
2976     _install_utrap;
2977     nop;
2978     _Q_cplx_div;
2979     _Q_cplx_div_ix;
2980     _Q_cplx_div_rx;
2981     _Q_cplx_lr_div;
2982     _Q_cplx_lr_div_ix;
2983     _Q_cplx_lr_div_rx;
2984     _Q_cplx_lr_mul;
2985     _Q_cplx_mul;
2986     _QgetRD;
2987     _xregs_clrptr;
2988 $endif

2990 $if sparc32
2991     __ashldi3;
2992     __ashrdi3;
2993     __cerror64;
2994     __cmpdi2;
2995     __floatdidf;
2996     __floatdisf;
2997     __floatundidf;
2998     __floatundisf;
2999     __lshrdi3;
3000     __muldi3;
3001     __ucmpdi2;
3002 $endif

3004 $if _x86
3005     _D_cplx_lr_div;
3006     _D_cplx_lr_div_ix;
3007     _D_cplx_lr_div_rx;
3008     _F_cplx_lr_div;
3009     _F_cplx_lr_div_ix;
3010     _F_cplx_lr_div_rx;
3011     __fltrounds;
3012     sysi86;
3013     __sysi86;
3014     _X_cplx_div;
3015     _X_cplx_div_ix;
3016     _X_cplx_div_rx;
3017     _X_cplx_lr_div;
3018     _X_cplx_lr_div_ix;
3019     _X_cplx_lr_div_rx;
3020     _X_cplx_mul;
3021     __xgetRD;
3022     __xtol;
3023     __xtoll;
3024     __xtoul;
3025     __xtoull;

```

```

3026 $endif

3028 $if i386
3029     __divrem64;
3030     __tls_get_addr;
3031     __udivrem64;
3032 $endif

3034 # The following functions should not be exported from libc,
3035 # but /lib/libm.so.2, some older versions of the Studio
3036 # compiler/debugger components, and some ancient programs
3037 # found in /usr/dist reference them. When we no longer
3038 # care about these old and broken binary objects, these
3039 # symbols should be deleted.
3040     _brk { FLAGS = NODYNSORT ; };
3041     _cond_broadcast { FLAGS = NODYNSORT ; };
3042     _cond_init { FLAGS = NODYNSORT ; };
3043     _cond_signal { FLAGS = NODYNSORT ; };
3044     _cond_wait { FLAGS = NODYNSORT ; };
3045     _ecvt { FLAGS = NODYNSORT ; };
3046     _fcvt { FLAGS = NODYNSORT ; };
3047     _getc_unlocked { FLAGS = NODYNSORT ; };
3048     _llseek { FLAGS = NODYNSORT ; };
3049     _pthread_attr_getdetachstate { FLAGS = NODYNSORT ; };
3050     _pthread_attr_getinheritsched { FLAGS = NODYNSORT ; };
3051     _pthread_attr_getschedparam { FLAGS = NODYNSORT ; };
3052     _pthread_attr_getschedpolicy { FLAGS = NODYNSORT ; };
3053     _pthread_attr_getscope { FLAGS = NODYNSORT ; };
3054     _pthread_attr_getstackaddr { FLAGS = NODYNSORT ; };
3055     _pthread_attr_getstacksize { FLAGS = NODYNSORT ; };
3056     _pthread_attr_init { FLAGS = NODYNSORT ; };
3057     _pthread_condattr_getpshared { FLAGS = NODYNSORT ; };
3058     _pthread_condattr_init { FLAGS = NODYNSORT ; };
3059     _pthread_cond_init { FLAGS = NODYNSORT ; };
3060     _pthread_create { FLAGS = NODYNSORT ; };
3061     _pthread_getschedparam { FLAGS = NODYNSORT ; };
3062     _pthread_join { FLAGS = NODYNSORT ; };
3063     _pthread_key_create { FLAGS = NODYNSORT ; };
3064     _pthread_mutexattr_getprioceiling { FLAGS = NODYNSORT ; };
3065     _pthread_mutexattr_getprotocol { FLAGS = NODYNSORT ; };
3066     _pthread_mutexattr_getpshared { FLAGS = NODYNSORT ; };
3067     _pthread_mutexattr_init { FLAGS = NODYNSORT ; };
3068     _pthread_mutex_getprioceiling { FLAGS = NODYNSORT ; };
3069     _pthread_mutex_init { FLAGS = NODYNSORT ; };
3070     _pthread_sigmask { FLAGS = NODYNSORT ; };
3071     _rwlock_init { FLAGS = NODYNSORT ; };
3072     _rw_rdlock { FLAGS = NODYNSORT ; };
3073     _rw_unlock { FLAGS = NODYNSORT ; };
3074     _rw_wllock { FLAGS = NODYNSORT ; };
3075     _sbrk_unlocked { FLAGS = NODYNSORT ; };
3076     _select { FLAGS = NODYNSORT ; };
3077     _sema_init { FLAGS = NODYNSORT ; };
3078     _sema_post { FLAGS = NODYNSORT ; };
3079     _sema_trywait { FLAGS = NODYNSORT ; };
3080     _sema_wait { FLAGS = NODYNSORT ; };
3081     _sysfs { FLAGS = NODYNSORT ; };
3082     _thr_create { FLAGS = NODYNSORT ; };
3083     _thr_exit { FLAGS = NODYNSORT ; };
3084     _thr_getprio { FLAGS = NODYNSORT ; };
3085     _thr_getspecific { FLAGS = NODYNSORT ; };
3086     _thr_join { FLAGS = NODYNSORT ; };
3087     _thr_keycreate { FLAGS = NODYNSORT ; };
3088     _thr_kill { FLAGS = NODYNSORT ; };
3089     _thr_main { FLAGS = NODYNSORT ; };
3090     _thr_self { FLAGS = NODYNSORT ; };
3091     _thr_setspecific { FLAGS = NODYNSORT ; };

```

```
3092     _thr_sigsetmask      { FLAGS = NODYNSORT };
3093     _thr_stksegment        { FLAGS = NODYNSORT };
3094     _ungetc_unlocked      { FLAGS = NODYNSORT };

3096     local:
3097     __imax_lldiv           { FLAGS = NODYNSORT };
3098     __ti_thr_self         { FLAGS = NODYNSORT };
3099     *;

3101 $if lf64
3102     __seekdir64           { FLAGS = NODYNSORT };
3103     __telldir64           { FLAGS = NODYNSORT };
3104 $endif

3106 $if _sparc
3107     __cerror              { FLAGS = NODYNSORT };
3108 $endif

3110 $if sparc32
3111     __cerror64            { FLAGS = NODYNSORT };
3112 $endif

3114 $if sparcv9
3115     __cleanup             { FLAGS = NODYNSORT };
3116 $endif

3118 $if i386
3119     __syscall6            { FLAGS = NODYNSORT };
3120     __systemcall6         { FLAGS = NODYNSORT };
3121 $endif

3123 $if amd64
3124     __tls_get_addr        { FLAGS = NODYNSORT };
3125 $endif
3126 };
_____unchanged_portion_omitted_____
```

```

*****
25709 Sat Jun 1 10:02:51 2013
new/usr/src/man/man1/ps.1
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
1  \" te
2 .\" Copyright 1989 AT&T
3 .\" Copyright (c) 2009, Sun Microsystems, Inc. All Rights Reserved
4 .\" Copyright (c) 2012, Joyent, Inc. All Rights Reserved
5 .\" Copyright (c) 2013 Gary Mills
6 .\" Portions Copyright (c) 1992, X/Open Company Limited All Rights Reserved
7 .\" Sun Microsystems, Inc. gratefully acknowledges The Open Group for permission
8 .\" The Institute of Electrical and Electronics Engineers and The Open Group, ha
9 .\" are reprinted and reproduced in electronic form in the Sun OS Reference Manu
10 .\" and Electronics Engineers, Inc and The Open Group. In the event of any discr
11 .\" This notice shall appear on any product containing this material.
12 .\" The contents of this file are subject to the terms of the Common Development
13 .\" See the License for the specific language governing permissions and limitat
14 .\" the fields enclosed by brackets \"[]\" replaced with your own identifying info
15 .TH PS 1 \"Apr 16, 2013\"
14 .TH PS 1 \"Aug 16, 2009\"
16 .SH NAME
17 ps \- report process status
18 .SH SYNOPSIS
19 .LP
20 .nf
21 \fBps\fR [\fB-aAcdefjHLLPWyZ\fR] [\fB-g\fR \fIgrplist\fR] [\fB-h\fR \fIlgprlist\
20 \fBps\fR [\fB-aAcdefjHLLPyZ\fR] [\fB-g\fR \fIgrplist\fR] [\fB-h\fR \fIlgprlist\
22 [\fB-n\fR \fIinamelist\fR] [\fB-o\fR \fIiformat\fR]... [\fB-p\fR \fIproclist\
23 [\fB-s\fR \fIisidlist\fR] [\fB-t\fR \fIterm\fR] [\fB-u\fR \fIuidlist\fR] [\f
24 [\fB-G\fR \fIgidlist\fR] [\fB-z\fR \fIizonelist\fR]
25 .fi

27 .SH DESCRIPTION
28 .sp
29 .LP
30 The \fBps\fR command prints information about active processes. Without
31 options, \fBps\fR prints information about processes that have the same
32 effective user \fBUID\fR and the same controlling terminal as the invoker. The
33 output contains only the process \fBUID\fR, terminal identifier, cumulative
34 execution time, and the command name. Otherwise, the information that is
35 displayed is controlled by the options.
36 .sp
37 .LP
38 Some options accept lists as arguments. Items in a list can be either separated
39 by commas or else enclosed in quotes and separated by commas or spaces. Values
40 for \fIproclist\fR and \fIgrplist\fR must be numeric.
41 .SH OPTIONS
42 .sp
43 .LP
44 The following options are supported:
45 .sp
46 .ne 2
47 .na
48 \fBfb-a\fR
49 .ad
50 .RS 15n
51 Lists information about \fBa\fRll processes most frequently requested: all
52 those except session leaders and processes not associated with a terminal.
53 .RE

55 .sp
56 .ne 2
57 .na
58 \fBfb-A\fR

```

```

59 .ad
60 .RS 15n
61 Lists information for all processes. Identical to \fB-e\fR, below.
62 .RE

64 .sp
65 .ne 2
66 .na
67 \fBfb-c\fR
68 .ad
69 .RS 15n
70 Prints information in a format that reflects scheduler properties as described
71 in \fBprioctl\fR(1). The \fB-c\fR option affects the output of the \fB-f\fR
72 and \fB-l\fR options, as described below.
73 .RE

75 .sp
76 .ne 2
77 .na
78 \fBfb-d\fR
79 .ad
80 .RS 15n
81 Lists information about all processes except session leaders.
82 .RE

84 .sp
85 .ne 2
86 .na
87 \fBfb-e\fR
88 .ad
89 .RS 15n
90 Lists information about \fBe\fRvery process now running.
91 .sp
92 When the \fB-e\fR option is specified, options \fB-z\fR, \fB-t\fR, \fB-u\fR,
93 \fB-U\fR, \fB-g\fR, \fB-G\fR, \fB-p\fR, \fB-g\fR, \fB-s\fR and \fB-a\fR options
94 have no effect.
95 .RE

97 .sp
98 .ne 2
99 .na
100 \fBfb-f\fR
101 .ad
102 .RS 15n
103 Generates a \fBfb-f\fRull listing. (See below for significance of columns in a
104 full listing.)
105 .RE

107 .sp
108 .ne 2
109 .na
110 \fBfb-g\fR \fIgrplist\fR
111 .ad
112 .RS 15n
113 Lists only process data whose group leader's \fBUID\fR number(s) appears in
114 \fIgrplist\fR. (A group leader is a process whose process \fBUID\fR number is
115 identical to its process group \fBUID\fR number.)
116 .RE

118 .sp
119 .ne 2
120 .na
121 \fBfb-G\fR \fIgidlist\fR
122 .ad
123 .RS 15n
124 Lists information for processes whose real group ID numbers are given in

```

```

125 \fIgidlist\fR. The \fIgidlist\fR must be a single argument in the form of a
126 blank- or comma-separated list.
127 .RE

129 .sp
130 .ne 2
131 .na
132 \fB\fB-h\fR \fIilgrplist\fR\fR
133 .ad
134 .RS 15n
135 Lists only the processes homed to the specified \fIilgrplist\fR. Nothing is
136 listed for any invalid group specified in \fIilgrplist\fR.
137 .RE

139 .sp
140 .ne 2
141 .na
142 \fB\fB-H\fR\fR
143 .ad
144 .RS 15n
145 Prints the home lgroup of the process under an additional column header, LGRP.
146 .RE

148 .sp
149 .ne 2
150 .na
151 \fB\fB-j\fR\fR
152 .ad
153 .RS 15n
154 Prints session \fBID\fR and process group \fBID\fR.
155 .RE

157 .sp
158 .ne 2
159 .na
160 \fB\fB-l\fR\fR
161 .ad
162 .RS 15n
163 Generates a \fBl\fRong listing. (See below.)
164 .RE

166 .sp
167 .ne 2
168 .na
169 \fB\fB-L\fR\fR
170 .ad
171 .RS 15n
172 Prints information about each light weight process (\fIlwp\fR) in each selected
173 process. (See below.)
174 .RE

176 .sp
177 .ne 2
178 .na
179 \fB\fB-n\fR \fInamelist\fR\fR
180 .ad
181 .RS 15n
182 Specifies the name of an alternative system \fInamelist\fR file in place of the
183 default. This option is accepted for compatibility, but is ignored.
184 .RE

186 .sp
187 .ne 2
188 .na
189 \fB\fB-o\fR \fIifformat\fR\fR
190 .ad

```

```

191 .RS 15n
192 Prints information according to the format specification given in \fIifformat\fR.
193 This is fully described in \fBDISPLAY FORMATS\fR. Multiple \fB-o\fR options can
194 be specified; the format specification is interpreted as the
195 space-character-separated concatenation of all the \fIifformat\fR
196 option-arguments.
197 .RE

199 .sp
200 .ne 2
201 .na
202 \fB\fB-p\fR \fIiproclist\fR\fR
203 .ad
204 .RS 15n
205 Lists only process data whose process \fBID\fR numbers are given in
206 \fIiproclist\fR.
207 .RE

209 .sp
210 .ne 2
211 .na
212 \fB\fB-P\fR\fR
213 .ad
214 .RS 15n
215 Prints the number of the processor to which the process or lwp is bound, if
216 any, under an additional column header, \fBPSR\fR.
217 .RE

219 .sp
220 .ne 2
221 .na
222 \fB\fB-s\fR \fIisidlist\fR\fR
223 .ad
224 .RS 15n
225 Lists information on all session leaders whose \fBID\fRs appear in
226 \fIisidlist\fR.
227 .RE

229 .sp
230 .ne 2
231 .na
232 \fB\fB-t\fR \fIitem\fR\fR
233 .ad
234 .RS 15n
235 Lists only process data associated with \fIitem\fR. Terminal identifiers are
236 specified as a device file name, and an identifier. For example, \fBterm/a\fR,
237 or \fBpts/0\fR.
238 .RE

240 .sp
241 .ne 2
242 .na
243 \fB\fB-u\fR \fIuidlist\fR\fR
244 .ad
245 .RS 15n
246 Lists only process data whose effective user \fBID\fR number or login name is
247 given in \fIuidlist\fR. In the listing, the numerical user \fBID\fR is printed
248 unless you give the \fB-f\fR option, which prints the login name.
249 .RE

251 .sp
252 .ne 2
253 .na
254 \fB\fB-U\fR \fIuidlist\fR\fR
255 .ad
256 .RS 15n

```

257 Lists information for processes whose real user \fBID\fR numbers or login names  
 258 are given in \fIuidlist\fR. The \fIuidlist\fR must be a single argument in the  
 259 form of a blank- or comma-separated list.  
 260 .RE

262 .sp  
 263 .ne 2  
 264 .na  
 265 \fB\fB-W\fR\fR  
 266 .ad  
 267 .RS 15n  
 268 Truncate long names even when \fBps\fR would normally print them  
 269 in full.  
 270 A trailing asterisk marks a long name that has been truncated  
 271 to fit the column.  
 272 .RE

274 .sp  
 275 .ne 2  
 276 .na  
 277 \fB\fB-y\fR\fR  
 278 .ad  
 279 .RS 15n  
 280 Under a long listing (\fB-l\fR), omits the obsolete \fBf\fR and \fBADDR\fR  
 281 columns and includes an \fBRSS\fR column to report the resident set size of the  
 282 process. Under the \fB-y\fR option, both \fBRSS\fR and \fBBSZ\fR (see below) is  
 283 reported in units of kilobytes instead of pages.  
 284 .RE

286 .sp  
 287 .ne 2  
 288 .na  
 289 \fB\fB-z\fR \fIzonelist\fR  
 290 .ad  
 291 .RS 15n  
 292 Lists only processes in the specified zones. Zones can be specified either by  
 293 name or ID. This option is only useful when executed in the global zone.  
 294 .RE

296 .sp  
 297 .ne 2  
 298 .na  
 299 \fB\fB-Z\fR\fR  
 300 .ad  
 301 .RS 15n  
 302 Prints the name of the zone with which the process is associated under an  
 303 additional column header, \fBZONE\fR. The \fBZONE\fR column width is limited to  
 304 8 characters. Use \fBps\fR \fB-eZ\fR for a quick way to see information about  
 305 every process now running along with the associated zone name. Use  
 306 .sp  
 307 .in +2  
 308 .nf  
 309 ps -eo zone,uid,pid,ppid,time,comm,...  
 310 .fi  
 311 .in -2  
 312 .sp

314 to see zone names wider than 8 characters.  
 315 .RE

317 .sp  
 318 .LP  
 319 Many of the options shown are used to select processes to list. If any are  
 320 specified, the default list is ignored and \fBps\fR selects the processes  
 321 represented by the inclusive OR of all the selection-criteria options.  
 322 .SH DISPLAY FORMATS

323 .sp  
 324 .LP  
 325 Under the \fB-f\fR option, \fBps\fR tries to determine the command name and  
 326 arguments given when the process was created by examining the user block.  
 327 Failing this, the command name is printed, as it would have appeared without  
 328 the \fB-f\fR option, in square brackets.  
 329 .sp  
 330 .LP  
 331 The column headings and the meaning of the columns in a \fBps\fR listing are  
 332 given below; the letters \fBf\fR and \fBl\fR indicate the option (\fBf\fR or  
 333 \fBl\fR, respectively) that causes the corresponding heading to appear;  
 334 \fBall\fR means that the heading always appears. \fBNote:\fR These two options  
 335 determine only what information is provided for a process; they do not  
 336 determine which processes are listed.  
 337 .sp  
 338 .ne 2  
 339 .na  
 340 \fB\fBf\fR(1)\fR  
 341 .ad  
 342 .RS 14n  
 343 Flags (hexadecimal and additive) associated with the process. These flags are  
 344 available for historical purposes; no meaning should be currently ascribed to  
 345 them.  
 346 .RE

348 .sp  
 349 .ne 2  
 350 .na  
 351 \fB\fBfS\fR(1)\fR  
 352 .ad  
 353 .RS 14n  
 354 The state of the process:  
 355 .sp  
 356 .ne 2  
 357 .na  
 358 \fBfBO\fR  
 359 .ad  
 360 .RS 5n  
 361 Process is running on a processor.  
 362 .RE

364 .sp  
 365 .ne 2  
 366 .na  
 367 \fBfBS\fR  
 368 .ad  
 369 .RS 5n  
 370 Sleeping: process is waiting for an event to complete.  
 371 .RE

373 .sp  
 374 .ne 2  
 375 .na  
 376 \fBfBR\fR  
 377 .ad  
 378 .RS 5n  
 379 Runnable: process is on run queue.  
 380 .RE

382 .sp  
 383 .ne 2  
 384 .na  
 385 \fBfBT\fR  
 386 .ad  
 387 .RS 5n  
 388 Process is stopped, either by a job control signal or because it is being

```

389 traced.
390 .RE

392 .sp
393 .ne 2
394 .na
395 \fB\fR\fr
396 .ad
397 .RS 5n
398 Waiting: process is waiting for CPU usage to drop to the CPU-caps enforced
399 limits.
400 .RE

402 .sp
403 .ne 2
404 .na
405 \fBZ\fr
406 .ad
407 .RS 5n
408 Zombie state: process terminated and parent not waiting.
409 .RE

411 .RE

413 .sp
414 .ne 2
415 .na
416 \fB\fR\fr (f,l)\fr
417 .ad
418 .RS 14n
419 The effective user \fBID\fr number of the process (the login name is printed
420 under the \fB-f\fr option).
421 A trailing asterisk marks a long name that has been truncated
422 to fit the column.
423 .RE

425 .sp
426 .ne 2
427 .na
428 \fB\fR\fr(all)\fr
429 .ad
430 .RS 14n
431 The process \fBID\fr of the process (this datum is necessary in order to kill a
432 process).
433 .RE

435 .sp
436 .ne 2
437 .na
438 \fB\fR\fr(f,l)\fr
439 .ad
440 .RS 14n
441 The process \fBID\fr of the parent process.
442 .RE

444 .sp
445 .ne 2
446 .na
447 \fB\fR\fr(f,l)\fr
448 .ad
449 .RS 14n
450 Processor utilization for scheduling (obsolete). Not printed when the \fB-c\fr
451 option is used.
452 .RE

454 .sp

```

```

455 .ne 2
456 .na
457 \fB\fR\fr(f,l)\fr
458 .ad
459 .RS 14n
460 Scheduling class. Printed only when the \fB-c\fr option is used.
461 .RE

463 .sp
464 .ne 2
465 .na
466 \fB\fR\fr(l)\fr
467 .ad
468 .RS 14n
469 The priority of the process. Without the \fB-c\fr option, higher numbers mean
470 lower priority. With the \fB-c\fr option, higher numbers mean higher priority.
471 .RE

473 .sp
474 .ne 2
475 .na
476 \fB\fR\fr(l)\fr
477 .ad
478 .RS 14n
479 Nice value, used in priority computation. Not printed when the \fB-c\fr option
480 is used. Only processes in the certain scheduling classes have a nice value.
481 .RE

483 .sp
484 .ne 2
485 .na
486 \fB\fR\fr(l)\fr
487 .ad
488 .RS 14n
489 The memory address of the process.
490 .RE

492 .sp
493 .ne 2
494 .na
495 \fB\fR\fr(l)\fr
496 .ad
497 .RS 14n
498 The total size of the process in virtual memory, including all mapped files and
499 devices, in pages. See \fBpagesize\fr(l).
500 .RE

502 .sp
503 .ne 2
504 .na
505 \fB\fR\fr(l)\fr
506 .ad
507 .RS 14n
508 The address of an event for which the process is sleeping (if blank, the
509 process is running).
510 .RE

512 .sp
513 .ne 2
514 .na
515 \fB\fR\fr(f)\fr
516 .ad
517 .RS 14n
518 The starting time of the process, given in hours, minutes, and seconds. (A
519 process begun more than twenty-four hours before the \fBps\fr inquiry is
520 executed is given in months and days.)

```



```

521 .RE

523 .sp
524 .ne 2
525 .na
526 \fB\fBTty\fR(all)\fR
527 .ad
528 .RS 14n
529 The controlling terminal for the process (the message, \fB?\fR, is printed when
530 there is no controlling terminal).
531 .RE

533 .sp
534 .ne 2
535 .na
536 \fB\fBTime\fR(all)\fR
537 .ad
538 .RS 14n
539 The cumulative execution time for the process.
540 .RE

542 .sp
543 .ne 2
544 .na
545 \fB\fBLTime\fR(all)\fR
546 .ad
547 .RS 14n
548 The execution time for the lwp being reported.
549 .RE

551 .sp
552 .ne 2
553 .na
554 \fB\fBCmd\fR(all)\fR
555 .ad
556 .RS 14n
557 The command name (the full command name and its arguments, up to a limit of 80
558 characters, are printed under the \fB-f\fR option).
559 .RE

561 .sp
562 .LP
563 The following two additional columns are printed when the \fB-j\fR option is
564 specified:
565 .sp
566 .ne 2
567 .na
568 \fB\fBPGID\fR\fR
569 .ad
570 .RS 8n
571 The process ID of the process group leader.
572 .RE

574 .sp
575 .ne 2
576 .na
577 \fB\fBSID\fR\fR
578 .ad
579 .RS 8n
580 The process ID of the session leader.
581 .RE

583 .sp
584 .LP
585 The following two additional columns are printed when the \fB-L\fR option is
586 specified:

```

```

587 .sp
588 .ne 2
589 .na
590 \fB\fBLWP\fR\fR
591 .ad
592 .RS 8n
593 The lwp ID of the lwp being reported.
594 .RE

596 .sp
597 .ne 2
598 .na
599 \fB\fBNLWP\fR\fR
600 .ad
601 .RS 8n
602 The number of lwps in the process (if \fB-f\fR is also specified).
603 .RE

605 .sp
606 .LP
607 Under the \fB-L\fR option, one line is printed for each lwp in the process and
608 the time-reporting fields \fB\fBTime\fR and \fB\fBLTime\fR show the values for the
609 lwp, not the process. A traditional single-threaded process contains only one
610 lwp.
611 .sp
612 .LP
613 A process that has exited and has a parent, but has not yet been waited for by
614 the parent, is marked \fB<defunct>\fR&.
615 .SS "\fB-o\fR format"
616 .sp
617 .LP
618 The \fB-o\fR option allows the output format to be specified under user
619 control.
620 .sp
621 .LP
622 The format specification must be a list of names presented as a single
623 argument, blank- or comma-separated. Each variable has a default header. The
624 default header can be overridden by appending an equals sign and the new text
625 of the header. The rest of the characters in the argument is used as the header
626 text. The fields specified are written in the order specified on the command
627 line, and should be arranged in columns in the output. The field widths are
628 selected by the system to be at least as wide as the header text (default or
629 overridden value). If the header text is null, such as \fB-o\fR \fIuser=,\fR
630 the field width is at least as wide as the default header text.
631 Long names are not truncated in this mode.
632 If all header text fields are null, no header line is written.
633 the field width is at least as wide as the default header text. If all header
634 text fields are null, no header line is written.
635 .sp
636 .LP
637 The following names are recognized in the POSIX locale:
638 .sp
639 .ne 2
640 .na
641 \fB\fBuser\fR\fR
642 .ad
643 .RS 10n
644 The effective user \fB\fBID\fR of the process. This is the textual user \fB\fBID\fR,
645 if it can be obtained and the field width permits, or a decimal representation
646 otherwise.
647 .RE

647 .sp
648 .ne 2
649 .na
650 \fB\fBouser\fR\fR

```

```

651 .ad
652 .RS 10n
653 The real user \fBID\fR of the process. This is the textual user \fBID\fR, if it
654 can be obtained and the field width permits, or a decimal representation
655 otherwise.
656 .RE

658 .sp
659 .ne 2
660 .na
661 \fB\fBgroup\fR\fR
662 .ad
663 .RS 10n
664 The effective group \fBID\fR of the process. This is the textual group
665 \fBID,\fR if it can be obtained and the field width permits, or a decimal
666 representation otherwise.
667 .RE

669 .sp
670 .ne 2
671 .na
672 \fB\fBrgroup\fR\fR
673 .ad
674 .RS 10n
675 The real group \fBID\fR of the process. This is the textual group \fBID,\fR if
676 it can be obtained and the field width permits, or a decimal representation
677 otherwise.
678 .RE

680 .sp
681 .ne 2
682 .na
683 \fB\fBpid\fR\fR
684 .ad
685 .RS 10n
686 The decimal value of the process \fBID\fR.
687 .RE

689 .sp
690 .ne 2
691 .na
692 \fB\fBppid\fR\fR
693 .ad
694 .RS 10n
695 The decimal value of the parent process \fBID\fR.
696 .RE

698 .sp
699 .ne 2
700 .na
701 \fB\fBpgid\fR\fR
702 .ad
703 .RS 10n
704 The decimal value of the process group \fBID.\fR
705 .RE

707 .sp
708 .ne 2
709 .na
710 \fB\fBpcpu\fR\fR
711 .ad
712 .RS 10n
713 The ratio of CPU time used recently to CPU time available in the same period,
714 expressed as a percentage. The meaning of ``recently'' in this context is
715 unspecified. The CPU time available is determined in an unspecified manner.
716 .RE

```

```

718 .sp
719 .ne 2
720 .na
721 \fB\fBvsz\fR\fR
722 .ad
723 .RS 10n
724 The total size of the process in virtual memory, in kilobytes.
725 .RE

727 .sp
728 .ne 2
729 .na
730 \fB\fBnice\fR\fR
731 .ad
732 .RS 10n
733 The decimal value of the system scheduling priority of the process. See
734 \fBnice\fR(1).
735 .RE

737 .sp
738 .ne 2
739 .na
740 \fB\fBetime\fR\fR
741 .ad
742 .RS 10n
743 In the POSIX locale, the elapsed time since the process was started, in the
744 form:
745 .sp
746 \fB[\fR\fR\fR-\fB]\fR\fR\fR:\fB]\fR\fR\fR:\fR\fR\fR
747 .sp
748 where
749 .sp
750 .ne 2
751 .na
752 \fB\fR\fR\fR
753 .ad
754 .RS 6n
755 is the number of days
756 .RE

758 .sp
759 .ne 2
760 .na
761 \fB\fR\fR\fR
762 .ad
763 .RS 6n
764 is the number of hours
765 .RE

767 .sp
768 .ne 2
769 .na
770 \fB\fR\fR\fR
771 .ad
772 .RS 6n
773 is the number of minutes
774 .RE

776 .sp
777 .ne 2
778 .na
779 \fB\fR\fR\fR
780 .ad
781 .RS 6n
782 is the number of seconds

```

```

783 .RE

785 The \fidd\fR field is a decimal integer. The \fIhh\fR, \fImm\fR and \fIss\fR
786 fields is two-digit decimal integers padded on the left with zeros.
787 .RE

789 .sp
790 .ne 2
791 .na
792 \fB\fBtime\fR\fR
793 .ad
794 .RS 10n
795 In the POSIX locale, the cumulative CPU time of the process in the form:
796 .sp
797 \fB[\fR\fIdd\fR-\fB]\fR\fIhh\fR:\fImm\fR:\fIss\fR
798 .sp
799 The \fIdd\fR, \fIhh\fR, \fImm\fR, and \fIss\fR fields is as described in the
800 \fBetime\fR specifier.
801 .RE

803 .sp
804 .ne 2
805 .na
806 \fB\fBtty\fR\fR
807 .ad
808 .RS 10n
809 The name of the controlling terminal of the process (if any) in the same format
810 used by the \fBwho\fR(1) command.
811 .RE

813 .sp
814 .ne 2
815 .na
816 \fB\fBcomm\fR\fR
817 .ad
818 .RS 10n
819 The name of the command being executed (\fBargv[0]\fR value) as a string.
820 .RE

822 .sp
823 .ne 2
824 .na
825 \fB\fBargs\fR\fR
826 .ad
827 .RS 10n
828 The command with all its arguments as a string. The implementation might
829 truncate this value to the field width; it is implementation-dependent whether
830 any further truncation occurs. It is unspecified whether the string represented
831 is a version of the argument list as it was passed to the command when it
832 started, or is a version of the arguments as they might have been modified by
833 the application. Applications cannot depend on being able to modify their
834 argument list and having that modification be reflected in the output of
835 \fBps\fR. The Solaris implementation limits the string to 80 bytes; the string
836 is the version of the argument list as it was passed to the command when it
837 started.
838 .RE

840 .sp
841 .LP
842 The following names are recognized in the Solaris implementation:
843 .sp
844 .ne 2
845 .na
846 \fB\fBf\fR\fR
847 .ad
848 .RS 11n

```

```

849 Flags (hexadecimal and additive) associated with the process.
850 .RE

852 .sp
853 .ne 2
854 .na
855 \fB\fBs\fR\fR
856 .ad
857 .RS 11n
858 The state of the process.
859 .RE

861 .sp
862 .ne 2
863 .na
864 \fB\fBc\fR\fR
865 .ad
866 .RS 11n
867 Processor utilization for scheduling (obsolete).
868 .RE

870 .sp
871 .ne 2
872 .na
873 \fB\fBuid\fR\fR
874 .ad
875 .RS 11n
876 The effective user \fBID\fR number of the process as a decimal integer.
877 .RE

879 .sp
880 .ne 2
881 .na
882 \fB\fBuid\fR\fR
883 .ad
884 .RS 11n
885 The real user \fBID\fR number of the process as a decimal integer.
886 .RE

888 .sp
889 .ne 2
890 .na
891 \fB\fBgid\fR\fR
892 .ad
893 .RS 11n
894 The effective group \fBID\fR number of the process as a decimal integer.
895 .RE

897 .sp
898 .ne 2
899 .na
900 \fB\fBrgid\fR\fR
901 .ad
902 .RS 11n
903 The real group \fBID\fR number of the process as a decimal integer.
904 .RE

906 .sp
907 .ne 2
908 .na
909 \fB\fBprojid\fR\fR
910 .ad
911 .RS 11n
912 The project \fBID\fR number of the process as a decimal integer.
913 .RE

```

```

915 .sp
916 .ne 2
917 .na
918 \fB\fBproject\fR\fR
919 .ad
920 .RS 11n
921 The project \fBID\fR of the process as a textual value if that value can be
922 obtained; otherwise, as a decimal integer.
923 .RE

925 .sp
926 .ne 2
927 .na
928 \fB\fBzoneid\fR\fR
929 .ad
930 .RS 11n
931 The zone \fBID\fR number of the process as a decimal integer.
932 .RE

934 .sp
935 .ne 2
936 .na
937 \fB\fBzone\fR\fR
938 .ad
939 .RS 11n
940 The zone \fBID\fR of the process as a textual value if that value can be
941 obtained; otherwise, as a decimal integer.
942 .RE

944 .sp
945 .ne 2
946 .na
947 \fB\fBsid\fR\fR
948 .ad
949 .RS 11n
950 The process ID of the session leader.
951 .RE

953 .sp
954 .ne 2
955 .na
956 \fB\fBtaskid\fR\fR
957 .ad
958 .RS 11n
959 The task \fBID\fR of the process.
960 .RE

962 .sp
963 .ne 2
964 .na
965 \fB\fBclass\fR\fR
966 .ad
967 .RS 11n
968 The scheduling class of the process.
969 .RE

971 .sp
972 .ne 2
973 .na
974 \fB\fBpri\fR\fR
975 .ad
976 .RS 11n
977 The priority of the process. Higher numbers mean higher priority.
978 .RE

980 .sp

```

```

981 .ne 2
982 .na
983 \fB\fBopri\fR\fR
984 .ad
985 .RS 11n
986 The obsolete priority of the process. Lower numbers mean higher priority.
987 .RE

989 .sp
990 .ne 2
991 .na
992 \fB\fBlwp\fR\fR
993 .ad
994 .RS 11n
995 The decimal value of the lwp \fBID\fR. Requesting this formatting option causes
996 one line to be printed for each lwp in the process.
997 .RE

999 .sp
1000 .ne 2
1001 .na
1002 \fB\fBnlwp\fR\fR
1003 .ad
1004 .RS 11n
1005 The number of lwps in the process.
1006 .RE

1008 .sp
1009 .ne 2
1010 .na
1011 \fB\fBpsr\fR\fR
1012 .ad
1013 .RS 11n
1014 The number of the processor to which the process or lwp is bound.
1015 .RE

1017 .sp
1018 .ne 2
1019 .na
1020 \fB\fBpset\fR\fR
1021 .ad
1022 .RS 11n
1023 The \fBID\fR of the processor set to which the process or lwp is bound.
1024 .RE

1026 .sp
1027 .ne 2
1028 .na
1029 \fB\fBaddr\fR\fR
1030 .ad
1031 .RS 11n
1032 The memory address of the process.
1033 .RE

1035 .sp
1036 .ne 2
1037 .na
1038 \fB\fBosz\fR\fR
1039 .ad
1040 .RS 11n
1041 The total size of the process in virtual memory, in pages.
1042 .RE

1044 .sp
1045 .ne 2
1046 .na

```

```

1047 \fB\fBwchan\fR\fR
1048 .ad
1049 .RS 11n
1050 The address of an event for which the process is sleeping (if \f(mi, the process
1051 is running).
1052 .RE

1054 .sp
1055 .ne 2
1056 .na
1057 \fB\fBstime\fR\fR
1058 .ad
1059 .RS 11n
1060 The starting time or date of the process, printed with no blanks.
1061 .RE

1063 .sp
1064 .ne 2
1065 .na
1066 \fB\fBbrss\fR\fR
1067 .ad
1068 .RS 11n
1069 The resident set size of the process, in kilobytes. The \fBbrss\fR value
1070 reported by \fBps\fR is an estimate provided by \fBproc\fR(4) that might
1071 underestimate the actual resident set size. Users who wish to get more accurate
1072 usage information for capacity planning should use \fBpmap\fR(1) \fB-x\fR
1073 instead.
1074 .RE

1076 .sp
1077 .ne 2
1078 .na
1079 \fB\fBpmem\fR\fR
1080 .ad
1081 .RS 11n
1082 The ratio of the process's resident set size to the physical memory on the
1083 machine, expressed as a percentage.
1084 .RE

1086 .sp
1087 .ne 2
1088 .na
1089 \fB\fBfname\fR\fR
1090 .ad
1091 .RS 11n
1092 The first 8 bytes of the base name of the process's executable file.
1093 .RE

1095 .sp
1096 .ne 2
1097 .na
1098 \fB\fBctid\fR\fR
1099 .ad
1100 .RS 11n
1101 The contract ID of the process contract the process is a member of as a decimal
1102 integer.
1103 .RE

1105 .sp
1106 .ne 2
1107 .na
1108 \fB\fBlgrp\fR\fR
1109 .ad
1110 .RS 11n
1111 The home lgroup of the process.
1112 .RE

```

```

1114 .sp
1115 .ne 2
1116 .na
1117 \fB\fBdmodel\fR\fR
1118 .ad
1119 .RS 11n
1120 The data model of the process, printed in the same manner as via
1121 \fBpflags\fR(1). The currently supported data models are _ILP32 and _LP64.
1122 .RE

1124 .sp
1125 .LP
1126 Only \fBcomm\fR and \fBargs\fR are allowed to contain blank characters; all
1127 others, including the Solaris implementation variables, are not.
1128 .sp
1129 .LP
1130 The following table specifies the default header to be used in the POSIX locale
1131 corresponding to each format specifier.
1132 .sp

1134 .sp
1135 .TS
1136 box;
1137 c c c c
1138 c c c c .
1139 Format Default Format Default
1140 Specifier Header Specifier Header
1141 -
1142 args COMMAND ppid PPID
1143 comm COMMAND rgroup RGROUP
1144 etime ELAPSED ruser RUSER
1145 group GROUP time TIME
1146 nice NI tty TT
1147 pcpu %CPU user USER
1148 ppid PGID vsz VSZ
1149 pid PID
1150 .TE

1152 .sp
1153 .LP
1154 The following table lists the Solaris implementation format specifiers and the
1155 default header used with each.
1156 .sp

1158 .sp
1159 .TS
1160 box;
1161 c c c c
1162 c c c c .
1163 Format Default Format Default
1164 Specifier Header Specifier Header
1165 -
1166 addr ADDR projid PROJID
1167 c C project PROJECT
1168 class CLS psr PSR
1169 f F rgid RGID
1170 fname COMMAND rss RSS
1171 gid GID ruid RUID
1172 lgrp LGRP s S
1173 lwp LWP sid SID
1174 nlwp NLWP stime STIME
1175 opri PRI taskid TASKID
1176 osz SZ uid UID
1177 pmem %MEM wchan WCHAN
1178 pri PRI zone ZONE

```

```

1179 ctid      CTID      zoneid  ZONEID
1180 .TE

1182 .SH EXAMPLES
1183 .LP
1184 \fBExample 1 \fRUsing \fBps\fR Command
1185 .sp
1186 .LP
1187 The command:

1189 .sp
1190 .in +2
1191 .nf
1192 example% \fBps -o user,pid,ppid=MOM -o args\fR
1193 .fi
1194 .in -2
1195 .sp

1197 .sp
1198 .LP
1199 writes the following in the POSIX locale:

1201 .sp
1202 .in +2
1203 .nf
1204 USER  PID  MOM  COMMAND
1205 helene 34   12  ps -o uid,pid,ppid=MOM -o args
1206 .fi
1207 .in -2
1208 .sp

1210 .sp
1211 .LP
1212 The contents of the \fBCOMMAND\fR field need not be the same due to possible
1213 truncation.

1215 .SH ENVIRONMENT VARIABLES
1216 .sp
1217 .LP
1218 See \fBenviron\fR(5) for descriptions of the following environment variables
1219 that affect the execution of \fBps\fR: \fBBLANG\fR, \fBBLC_ALL\fR,
1220 \fBBLC_CTYPE\fR, \fBBLC_MESSAGES\fR, \fBBLC_TIME\fR, and \fBBLSPATH\fR.
1221 .sp
1222 .ne 2
1223 .na
1224 \fB\BCOLUMNS\fR
1225 .ad
1226 .RS 11n
1227 Override the system-selected horizontal screen size, used to determine the
1228 number of text columns to display.
1229 .RE

1231 .SH EXIT STATUS
1232 .sp
1233 .LP
1234 The following exit values are returned:
1235 .sp
1236 .ne 2
1237 .na
1238 \fB\B0\fR
1239 .ad
1240 .RS 6n
1241 Successful completion.
1242 .RE

1244 .sp

```

```

1245 .ne 2
1246 .na
1247 \fB\B>0\fR
1248 .ad
1249 .RS 6n
1250 An error occurred.
1251 .RE

1253 .SH FILES
1254 .sp
1255 .ne 2
1256 .na
1257 \fB\B/dev/pts/*\fR
1258 .ad
1259 .RS 15n

1261 .RE

1263 .sp
1264 .ne 2
1265 .na
1266 \fB\B/dev/term/*\fR
1267 .ad
1268 .RS 15n
1269 terminal ('tty') names searcher files
1270 .RE

1272 .sp
1273 .ne 2
1274 .na
1275 \fB\B/etc/passwd\fR
1276 .ad
1277 .RS 15n
1278 \fB\BUID\fR information supplier
1279 .RE

1281 .sp
1282 .ne 2
1283 .na
1284 \fB\B/proc/*\fR
1285 .ad
1286 .RS 15n
1287 process control files
1288 .RE

1290 .SH ATTRIBUTES
1291 .sp
1292 .LP
1293 See \fBattributes\fR(5) for descriptions of the following attributes:
1294 .sp

1296 .sp
1297 .TS
1298 box;
1299 c | c
1300 l | l .
1301 ATTRIBUTE TYPE  ATTRIBUTE VALUE
1302 _
1303 CSI             Enabled (see USAGE)
1304 _
1305 Interface Stability  Committed
1306 _
1307 Standard        See \fBstandards\fR(5).
1308 .TE

1310 .SH SEE ALSO

```

1311 .sp  
1312 .LP  
1313 \fBkill\fR(1), \fBlgrpinfo\fR(1), \fBnice\fR(1), \fBpagesize\fR(1),  
1314 \fBpmap\fR(1), \fBpricntl\fR(1), \fBwho\fR(1), \fBgetty\fR(1M), \fBproc\fR(4),  
1315 \fBttsrch\fR(4), \fBattributes\fR(5), \fBenviron\fR(5),  
1316 \fBresource\_controls\fR(5), \fBstandards\fR(5), \fBzones\fR(5)  
1317 .SH NOTES  
1318 .sp  
1319 .LP  
1320 Things can change while \fBps\fR is running. The snapshot it gives is true only  
1321 for a split-second, and it might not be accurate by the time you see it. Some  
1322 data printed for defunct processes is irrelevant.  
1323 .sp  
1324 .LP  
1325 If no options to select processes are specified, \fBps\fR reports all processes  
1326 associated with the controlling terminal. If there is no controlling terminal,  
1327 there is no report other than the header.  
1328 .sp  
1329 .LP  
1330 \fBps\fR \fB-ef\fR or \fBps\fR \fB-o\fR \fBstime\fR might not report the actual  
1331 start of a tty login session, but rather an earlier time, when a getty was last  
1332 respawned on the tty line.  
1333 .sp  
1334 .LP  
1335 \fBps\fR is \fBCSI\fR-enabled except for login names (usernames).

```

*****
16922 Sat Jun 1 10:02:51 2013
new/usr/src/man/man1m/prstat.1m
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
1 \" te
2 .\" Copyright (c) 2013 Gary Mills
3 .\" Copyright (c) 2006, 2009 Sun Microsystems, Inc. All Rights Reserved.
4 .\" The contents of this file are subject to the terms of the Common Development
5 .\" See the License for the specific language governing permissions and limitat
6 .\" the fields enclosed by brackets \"[]\" replaced with your own identifying info
7 .TH PRSTAT 1M \"Apr 15, 2013\"
8 .TH PRSTAT 1M \"Jun 25, 2009\"
9 .SH NAME
10 prstat \- report active process statistics
11 .SH SYNOPSIS
12 .nf
13 \fBprstat\fR [\fB-acHJLmRrtTvWZ\fR] [\fB-d\fR u | d] [\fB-C\fR \fIpsrsetlist\fR]
12 \fBprstat\fR [\fB-acHJLmRrtTv\fR] [\fB-d\fR u | d] [\fB-C\fR \fIpsrsetlist\fR] [
14 [\fB-j\fR \fIprojlist\fR] [\fB-k\fR \fItasklist\fR] [\fB-n\fR \fIntop\fR] \,
15 [\fB-p\fR \fIpidlist\fR] [\fB-P\fR \fIcpulist\fR] [\fB-s\fR \fIkey\fR | \fB
16 [\fB-u\fR \fIuidlist\fR] [\fB-U\fR \fIuidlist\fR] [\fB-z\fR \fIzoneidlist\
15 [\fB-u\fR \fIuidlist\fR] [\fB-U\fR \fIuidlist\fR] [\fB-z\fR \fIzoneidlist\
17 [\fIinterval\fR [\fIcount\fR]]
18 .fi

20 .SH DESCRIPTION
21 .sp
22 .LP
23 The \fBprstat\fR utility iteratively examines all active processes on the
24 system and reports statistics based on the selected output mode and sort order.
25 \fBprstat\fR provides options to examine only processes matching specified
26 \fBPID\fRs, \fBUID\fRs, zone \fBID\fRs, \fBCPU\fR \fBID\fRs, and processor set
27 \fBID\fRs.
28 .sp
29 .LP
30 The \fB-j\fR, \fB-k\fR, \fB-C\fR, \fB-p\fR, \fB-P\fR, \fB-u\fR, \fB-U\fR, and
31 \fB-z\fR options accept lists as arguments. Items in a list can be either
32 separated by commas or enclosed in quotes and separated by commas or spaces.
33 .sp
34 .LP
35 If you do not specify an option, \fBprstat\fR examines all processes and
36 reports statistics sorted by \fBCPU\fR usage.
37 .SH OPTIONS
38 .sp
39 .LP
40 The following options are supported:
41 .sp
42 .ne 2
43 .na
44 \fB-a\fR
45 .ad
46 .sp .6
47 .RS 4n
48 Report information about processes and users. In this mode \fBprstat\fR
49 displays separate reports about processes and users at the same time.
50 .RE

52 .sp
53 .ne 2
54 .na
55 \fB-c\fR
56 .ad
57 .sp .6

```

```

58 .RS 4n
59 Print new reports below previous reports instead of overprinting them.
60 Long names are not truncated in this mode.
61 .RE

63 .sp
64 .ne 2
65 .na
66 \fB-C\fR \fIpsrsetlist\fR
67 .ad
68 .sp .6
69 .RS 4n
70 Report only processes or lwps that are bound to processor sets in the given
71 list. Each processor set is identified by an integer as reported by
72 \fBpsrset\fR(1M). The load averages displayed are the sum of the load averages
73 of the specified processor sets (see \fBpsrset_getloadavg\fR(3C)). Processes with
74 one or more LWPs bound to processor sets in the given list are reported even
75 when the \fB-L\fR option is not used.
76 .RE

78 .sp
79 .ne 2
80 .na
81 \fB-d\fR \fBu | d\fR
82 .ad
83 .sp .6
84 .RS 4n
85 Specify \fBu\fR for a printed representation of the internal representation of
86 time. See \fBtime\fR(2). Specify \fBd\fR for standard date format. See
87 \fBdate\fR(1).
88 .RE

90 .sp
91 .ne 2
92 .na
93 \fB-h\fR \fIilgrplist\fR
94 .ad
95 .sp .6
96 .RS 4n
97 Report only processes or lwps whose home \fIilgroup\fR is in the given list of
98 \fIilgroups\fR. No processes or lwps will be listed for invalid \fIilgroups\fR.
99 .RE

101 .sp
102 .ne 2
103 .na
104 \fB-H\fR
105 .ad
106 .sp .6
107 .RS 4n
108 Report information about home \fIilgroup\fR. In this mode, \fBprstat\fR adds an
109 extra column showing process or lwps home \fIilgroup\fR with the header LGRP.
110 .RE

112 .sp
113 .ne 2
114 .na
115 \fB-j\fR \fIprojlist\fR
116 .ad
117 .sp .6
118 .RS 4n
119 Report only processes or lwps whose project \fBID\fR is in the given list. Each
120 project \fBID\fR can be specified as either a project name or a numerical
121 project \fBID\fR. See \fBproject\fR(4).
122 .RE

```



```

124 .sp
125 .ne 2
126 .na
127 \fB\fB-J\fR\fR
128 .ad
129 .sp .6
130 .RS 4n
131 Report information about processes and projects. In this mode \fBprstat\fR
132 displays separate reports about processes and projects at the same time.
133 A trailing asterisk marks a long name that has been truncated
134 to fit the column.
135 .RE

137 .sp
138 .ne 2
139 .na
140 \fB\fB-k\fR \fItasklist\fR
141 .ad
142 .sp .6
143 .RS 4n
144 Report only processes or lwps whose task \fBID\fR is in \fItasklist\fR.
145 .RE

147 .sp
148 .ne 2
149 .na
150 \fB\fB-L\fR\fR
151 .ad
152 .sp .6
153 .RS 4n
154 Report statistics for each light-weight process (\fBLOWP\fR). By default,
155 \fBprstat\fR reports only the number of \fBLOWP\fRs for each process.
156 .RE

158 .sp
159 .ne 2
160 .na
161 \fB\fB-m\fR\fR
162 .ad
163 .sp .6
164 .RS 4n
165 Report microstate process accounting information. In addition to all fields
166 listed in \fB-v\fR mode, this mode also includes the percentage of time the
167 process has spent processing system traps, text page faults, data page faults,
168 waiting for user locks and waiting for \fBCPU\fR (latency time).
169 .RE

171 .sp
172 .ne 2
173 .na
174 \fB\fB-n\fR \fIntop\fR[\fI,nbottom\fR]\fR
175 .ad
176 .sp .6
177 .RS 4n
178 Restrict number of output lines. The \fIntop\fR argument determines how many
179 lines of process or \fBlwp\fR statistics are reported, and the \fInbottom\fR
180 argument determines how many lines of user, task, or projects statistics are
181 reported if the \fB-a\fR, \fB-t\fR, \fB-T\fR, or \fB-J\fR options are
182 specified. By default, \fBprstat\fR displays as many lines of output that fit
183 in a window or terminal. When you specify the \fB-c\fR option or direct the
184 output to a file, the default values for \fBntop\fR and \fBnbottom\fR are
185 \fB15\fR and \fB5\fR.
186 .RE

188 .sp
189 .ne 2

```

```

190 .na
191 \fB\fB-p\fR \fIpidlist\fR\fR
192 .ad
193 .sp .6
194 .RS 4n
195 Report only processes whose process \fBID\fR is in the given list.
196 .RE

198 .sp
199 .ne 2
200 .na
201 \fB\fB-P\fR \fIcpulist\fR\fR
202 .ad
203 .sp .6
204 .RS 4n
205 Report only processes or \fBlwp\fRs which have most recently executed on a
206 \fBCPU\fR in the given list. Each \fBCPU\fR is identified by an integer as
207 reported by \fBpsrinfo\fR(1M).
208 .RE

210 .sp
211 .ne 2
212 .na
213 \fB\fB-R\fR\fR
214 .ad
215 .sp .6
216 .RS 4n
217 Put \fBprstat\fR in the real time scheduling class. When this option is used,
218 \fBprstat\fR is given priority over time-sharing and interactive processes.
219 This option is available only for superuser.
220 .RE

222 .sp
223 .ne 2
224 .na
225 \fB\fB-r\fR\fR
226 .ad
227 .sp .6
228 .RS 4n
229 Disable lookups for user names and project names. (Note that this does not
230 apply to lookups for the \fB-j\fR, \fB-u\fR, or \fB-U\fR options.)
231 .RE

233 .sp
234 .ne 2
235 .na
236 \fB\fB-s\fR \fIkey\fR\fR
237 .ad
238 .sp .6
239 .RS 4n
240 Sort output lines (that is, processes, \fBlwp\fRs, or users) by \fIkey\fR in
241 descending order. Only one \fIkey\fR can be used as an argument.
242 .sp
243 There are five possible key values:
244 .sp
245 .ne 2
246 .na
247 \fBcpu\fR
248 .ad
249 .sp .6
250 .RS 4n
251 Sort by process \fBCPU\fR usage. This is the default.
252 .RE

254 .sp
255 .ne 2

```

```

256 .na
257 \fBpri\fR
258 .ad
259 .sp .6
260 .RS 4n
261 Sort by process priority.
262 .RE

264 .sp
265 .ne 2
266 .na
267 \fBrsz\fR
268 .ad
269 .sp .6
270 .RS 4n
271 Sort by resident set size.
272 .RE

274 .sp
275 .ne 2
276 .na
277 \fBsize\fR
278 .ad
279 .sp .6
280 .RS 4n
281 Sort by size of process image.
282 .RE

284 .sp
285 .ne 2
286 .na
287 \fBtime\fR
288 .ad
289 .sp .6
290 .RS 4n
291 Sort by process execution time.
292 .RE

294 .RE

296 .sp
297 .ne 2
298 .na
299 \fB\fB-S\fR \fIkey\fR\fR
300 .ad
301 .sp .6
302 .RS 4n
303 Sort output lines by \fIkey\fR in ascending order. Possible \fIkey\fR values
304 are the same as for the \fB-s\fR option. See \fB-s\fR.
305 .RE

307 .sp
308 .ne 2
309 .na
310 \fB\fB-t\fR\fR
311 .ad
312 .sp .6
313 .RS 4n
314 Report total usage summary for each user. The summary includes the total number
315 of processes or \fBBLWP\fRs owned by the user, total size of process images,
316 total resident set size, total cpu time, and percentages of recent cpu time and
317 system memory.
318 .RE

320 .sp
321 .ne 2

```

```

322 .na
323 \fB\fB-T\fR\fR
324 .ad
325 .sp .6
326 .RS 4n
327 Report information about processes and tasks. In this mode \fBprstat\fR
328 displays separate reports about processes and tasks at the same time.
329 .RE

331 .sp
332 .ne 2
333 .na
334 \fB\fB-u\fR \fIeuidlist\fR\fR
335 .ad
336 .sp .6
337 .RS 4n
338 Report only processes whose effective user \fBID\fR is in the given list. Each
339 user \fBID\fR may be specified as either a login name or a numerical user
340 \fBID\fR.
341 .RE

343 .sp
344 .ne 2
345 .na
346 \fB\fB-U\fR \fIuidlis\fRt\fR
347 .ad
348 .sp .6
349 .RS 4n
350 Report only processes whose real user \fBID\fR is in the given list. Each user
351 \fBID\fR may be specified as either a login name or a numerical user \fBID\fR.
352 .RE

354 .sp
355 .ne 2
356 .na
357 \fB\fB-v\fR\fR
358 .ad
359 .sp .6
360 .RS 4n
361 Report verbose process usage. This output format includes the percentage of
362 time the process has spent in user mode, in system mode, and sleeping. It also
363 includes the number of voluntary and involuntary context switches, system calls
364 and the number of signals received. Statistics that are not reported are marked
365 with the \fB-\fR sign.
366 .RE

368 .sp
369 .ne 2
370 .na
371 \fB\fB-W\fR\fR
372 .ad
373 .sp .6
374 .RS 4n
375 Truncate long names even when \fBprstat\fR would normally print them
376 in full.
377 A trailing asterisk marks a long name that has been truncated
378 to fit the column.
379 .RE

381 .sp
382 .ne 2
383 .na
384 \fB\fB-z\fR \fIzoneidlist\fR\fR
385 .ad
386 .sp .6
387 .RS 4n

```

```

388 Report only processes or LWPs whose zone ID is in the given list. Each zone ID
389 can be specified as either a zone name or a numerical zone ID. See
390 \fBzones\fR(5).
391 .RE

393 .sp
394 .ne 2
395 .na
396 \fB\fB-Z\fR\fR
397 .ad
398 .sp .6
399 .RS 4n
400 Report information about processes and zones. In this mode, \fBprstat\fR
401 displays separate reports about processes and zones at the same time.
402 A trailing asterisk marks a long name that has been truncated
403 to fit the column.
404 .RE

406 .SH OUTPUT
407 .sp
408 .LP
409 The following list defines the column headings and the meanings of a
410 \fBprstat\fR report:
411 .sp
412 .ne 2
413 .na
414 \fBFPID\fR
415 .ad
416 .sp .6
417 .RS 4n
418 The process \fBID\fR of the process.
419 .RE

421 .sp
422 .ne 2
423 .na
424 \fBUSERNAME\fR
425 .ad
426 .sp .6
427 .RS 4n
428 The real user (login) name or real user \fBID\fR.
429 A trailing asterisk marks a long name that has been truncated
430 to fit the column.
431 .RE

433 .sp
434 .ne 2
435 .na
436 \fBFSWAP\fR
437 .ad
438 .sp .6
439 .RS 4n
440 The total virtual memory size of the process, including all mapped files and
441 devices, in kilobytes (\fBK\fR), megabytes (\fBM\fR), or gigabytes (\fBG\fR).
442 .RE

444 .sp
445 .ne 2
446 .na
447 \fBRSS\fR
448 .ad
449 .sp .6
450 .RS 4n
451 The resident set size of the process (\fBRSS\fR), in kilobytes (\fBK\fR),
452 megabytes (\fBM\fR), or gigabytes (\fBG\fR). The RSS value is an estimate
453 provided by \fBproc\fR(4) that might underestimate the actual resident set

```

```

454 size. Users who want to get more accurate usage information for capacity
455 planning should use the \fB-x\fR option to \fBpmap\fR(1) instead.
456 .RE

458 .sp
459 .ne 2
460 .na
461 \fBSTATE\fR
462 .ad
463 .sp .6
464 .RS 4n
465 The state of the process:
466 .sp
467 .ne 2
468 .na
469 \fBcpu\fIN\fR
470 .ad
471 .sp .6
472 .RS 4n
473 Process is running on \fBCPU\fR \fIN\fR.
474 .RE

476 .sp
477 .ne 2
478 .na
479 \fBsleeeep\fR
480 .ad
481 .sp .6
482 .RS 4n
483 Sleeping: process is waiting for an event to complete.
484 .RE

486 .sp
487 .ne 2
488 .na
489 \fBwait\fR
490 .ad
491 .sp .6
492 .RS 4n
493 Waiting: process is waiting for CPU usage to drop to the CPU-caps enforced
494 limits. See the description of \fBCPU-caps\fR in \fBresource_controls\fR(5).
495 .RE

497 .sp
498 .ne 2
499 .na
500 \fBrun\fR
501 .ad
502 .sp .6
503 .RS 4n
504 Runnable: process in on run queue.
505 .RE

507 .sp
508 .ne 2
509 .na
510 \fBzombie\fR
511 .ad
512 .sp .6
513 .RS 4n
514 Zombie state: process terminated and parent not waiting.
515 .RE

517 .sp
518 .ne 2
519 .na

```

```

520 \fBstop\fR
521 .ad
522 .sp .6
523 .RS 4n
524 Process is stopped.
525 .RE

527 .RE

529 .sp
530 .ne 2
531 .na
532 \fBPRI\fR
533 .ad
534 .sp .6
535 .RS 4n
536 The priority of the process. Larger numbers mean higher priority.
537 .RE

539 .sp
540 .ne 2
541 .na
542 \fBNICE\fR
543 .ad
544 .sp .6
545 .RS 4n
546 Nice value used in priority computation. Only processes in certain scheduling
547 classes have a nice value.
548 .RE

550 .sp
551 .ne 2
552 .na
553 \fBTIME\fR
554 .ad
555 .sp .6
556 .RS 4n
557 The cumulative execution time for the process.
558 .RE

560 .sp
561 .ne 2
562 .na
563 \fBCPU\fR
564 .ad
565 .sp .6
566 .RS 4n
567 The percentage of recent \fBCPU\fR time used by the process. If executing in a
568 non-global \fBzone\fR and the pools facility is active, the percentage will be
569 that of the processors in the processor set in use by the pool to which the
570 \fBzone\fR is bound.
571 .RE

573 .sp
574 .ne 2
575 .na
576 \fBPROCESS\fR
577 .ad
578 .sp .6
579 .RS 4n
580 The name of the process (name of executed file).
581 .RE

583 .sp
584 .ne 2
585 .na

```

```

586 \fBLWPID\fR
587 .ad
588 .sp .6
589 .RS 4n
590 The \fBlwp\fR \fBID\fR of the \fBlwp\fR being reported.
591 .RE

593 .sp
594 .ne 2
595 .na
596 \fBNLWP\fR
597 .ad
598 .sp .6
599 .RS 4n
600 The number of \fBlwp\fRs in the process.
601 .RE

603 .sp
604 .LP
605 With the some options, in addition to a number of the column headings shown
606 above, there are:
607 .sp
608 .ne 2
609 .na
610 \fBNPROC\fR
611 .ad
612 .sp .6
613 .RS 4n
614 Number of processes in a specified collection.
615 .RE

617 .sp
618 .ne 2
619 .na
620 \fBMEMORY\fR
621 .ad
622 .sp .6
623 .RS 4n
624 Percentage of memory used by a specified collection of processes.
625 .RE

627 .sp
628 .LP
629 The following columns are displayed when the \fB-v\fR or \fB-m\fR option is
630 specified
631 .sp
632 .ne 2
633 .na
634 \fBUSR\fR
635 .ad
636 .sp .6
637 .RS 4n
638 The percentage of time the process has spent in user mode.
639 .RE

641 .sp
642 .ne 2
643 .na
644 \fBSYS\fR
645 .ad
646 .sp .6
647 .RS 4n
648 The percentage of time the process has spent in system mode.
649 .RE

651 .sp

```

```

652 .ne 2
653 .na
654 \fBTRP\fR
655 .ad
656 .sp .6
657 .RS 4n
658 The percentage of time the process has spent in processing system traps.
659 .RE

661 .sp
662 .ne 2
663 .na
664 \fBTFLL\fR
665 .ad
666 .sp .6
667 .RS 4n
668 The percentage of time the process has spent processing text page faults.
669 .RE

671 .sp
672 .ne 2
673 .na
674 \fBDFLL\fR
675 .ad
676 .sp .6
677 .RS 4n
678 The percentage of time the process has spent processing data page faults.
679 .RE

681 .sp
682 .ne 2
683 .na
684 \fBLLCK\fR
685 .ad
686 .sp .6
687 .RS 4n
688 The percentage of time the process has spent waiting for user locks.
689 .RE

691 .sp
692 .ne 2
693 .na
694 \fBLLP\fR
695 .ad
696 .sp .6
697 .RS 4n
698 The percentage of time the process has spent sleeping.
699 .RE

701 .sp
702 .ne 2
703 .na
704 \fBLLAT\fR
705 .ad
706 .sp .6
707 .RS 4n
708 The percentage of time the process has spent waiting for CPU.
709 .RE

711 .sp
712 .ne 2
713 .na
714 \fBVVCX\fR
715 .ad
716 .sp .6
717 .RS 4n

```

```

718 The number of voluntary context switches.
719 .RE

721 .sp
722 .ne 2
723 .na
724 \fBICX\fR
725 .ad
726 .sp .6
727 .RS 4n
728 The number of involuntary context switches.
729 .RE

731 .sp
732 .ne 2
733 .na
734 \fBSCCL\fR
735 .ad
736 .sp .6
737 .RS 4n
738 The number of system calls.
739 .RE

741 .sp
742 .ne 2
743 .na
744 \fBSIG\fR
745 .ad
746 .sp .6
747 .RS 4n
748 The number of signals received.
749 .RE

751 .sp
752 .LP
753 Under the \fB-L\fR option, one line is printed for each \fBlwp\fR in the
754 process and some reporting fields show the values for the \fBlwp\fR, not the
755 process.
756 .sp
757 .LP
758 The following column is displayed when the \fB-H\fR option is specified:
759 .sp
760 .ne 2
761 .na
762 \fBLLGRP\fR
763 .ad
764 .sp .6
765 .RS 4n
766 The home \fIlgroupl\fR of the process or lwp.
767 .RE

769 .SH OPERANDS
770 .sp
771 .LP
772 The following operands are supported:
773 .sp
774 .ne 2
775 .na
776 \fBfIcount\fR
777 .ad
778 .sp .6
779 .RS 4n
780 Specifies the number of times that the statistics are repeated. By default,
781 \fBprstat\fR reports statistics until a termination signal is received.
782 .RE

```

```

784 .sp
785 .ne 2
786 .na
787 \fB\fIinterval\fR\fR
788 .ad
789 .sp .6
790 .RS 4n
791 Specifies the sampling interval in seconds; the default interval is \fB5\fR
792 seconds.
793 .RE

795 .SH EXAMPLES
796 .LP
797 \fBExample 1 \fRReporting the Five Most Active Super-User Processes
798 .sp
799 .LP
800 The following command reports the five most active super-user processes running
801 on \fBCPU1\fR and \fBCPU2\fR:

803 .sp
804 .in +2
805 .nf
806 example% prstat -u root -n 5 -P 1,2 1 1

808 PID USERNAME SWAP  RSS STATE  PRI  NICE      TIME  CPU PROCESS/LWP
809 306 root      3024K 1448K sleep  58    0  0:00.00 0.3% sendmail/1
810 102 root      1600K  592K sleep  59    0  0:00.00 0.1% in.rdisc/1
811 250 root      1000K  552K sleep  58    0  0:00.00 0.0% utmpd/1
812 288 root      1720K 1032K sleep  58    0  0:00.00 0.0% sac/1
813 1 root       744K  168K sleep  58    0  0:00.00 0.0% init/1
814 TOTAL:      25, load averages: 0.05, 0.08, 0.12
815 .fi
816 .in -2
817 .sp

819 .LP
820 \fBExample 2 \fRDisplaying Verbose Process Usage Information
821 .sp
822 .LP
823 The following command displays verbose process usage information about
824 processes with lowest resident set sizes owned by users \fBroot\fR and
825 \fBjohn\fR.

827 .sp
828 .in +2
829 .nf
830 example% prstat -S rss -n 5 -vc -u root,john

832 PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWP
833 1 root      0.0 0.0 - - - - 100 - 0 0 0 0 init/1
834 102 root    0.0 0.0 - - - - 100 - 0 0 3 0 in.rdisc/1
835 250 root    0.0 0.0 - - - - 100 - 0 0 0 0 utmpd/1
836 1185 john   0.0 0.0 - - - - 100 - 0 0 0 0 csh/1
837 240 root    0.0 0.0 - - - - 100 - 0 0 0 0 powerd/4
838 TOTAL:      71, load averages: 0.02, 0.04, 0.08

840 .fi
841 .in -2
842 .sp

844 .SH EXIT STATUS
845 .sp
846 .LP
847 The following exit values are returned:
848 .sp
849 .ne 2

```

```

850 .na
851 \fB\fB0\fR\fR
852 .ad
853 .sp .6
854 .RS 4n
855 Successful completion.
856 .RE

858 .sp
859 .ne 2
860 .na
861 \fB\fB1\fR\fR
862 .ad
863 .sp .6
864 .RS 4n
865 An error occurred.
866 .RE

868 .SH SEE ALSO
869 .sp
870 .LP
871 \fBdate\fR(1), \fBlgrpinfo\fR(1), \fBplgrp\fR(1), \fBproc\fR(1), \fBps\fR(1),
872 \fBtime\fR(2), \fBpsrinfo\fR(1M), \fBpsrset\fR(1M), \fBsar\fR(1M),
873 \fBpset_getloadavg\fR(3C), \fBproc\fR(4), \fBproject\fR(4),
874 \fBattributes\fR(5), \fBresource_controls\fR(5), \fBzones\fR(5)
875 .SH NOTES
876 .sp
877 .LP
878 The snapshot of system usage displayed by \fBprstat\fR is true only for a
879 split-second, and it may not be accurate by the time it is displayed. When the
880 \fB-m\fR option is specified, \fBprstat\fR tries to turn on microstate
881 accounting for each process; the original state is restored when \fBprstat\fR
882 exits. See \fBproc\fR(4) for additional information about the microstate
883 accounting facility.
884 .sp
885 .LP
886 The total memory size reported in the SWAP and RSS columns for groups of
887 processes can sometimes overestimate the actual amount of memory used by
888 processes with shared memory segments.

```

```

*****
12884 Sat Jun 1 10:02:51 2013
new/usr/src/man/man1m/useradd.1m
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
1 \" te
2 .\" Copyright (c) 2013 Gary Mills
3 .\" Copyright (c) 2008 Sun Microsystems, Inc. All Rights Reserved.
4 .\" Copyright 1989 AT&T
5 .\" The contents of this file are subject to the terms of the Common Development
6 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
7 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
8 .TH USERADD 1M \"Apr 16, 2013\"
9 .TH USERADD 1M \"Feb 19, 2008\"
10 .SH NAME
11 useradd \- administer a new user login on the system
12 .SH SYNOPSIS
13 .LP
14 .nf
15 \fBuseradd\fR [\fB-A\fR \fIauthorization\fR [, \fIauthorization...\fR]]
16 [\fB-b\fR \fIbase_dir\fR] [\fB-c\fR \fIcomment\fR] [\fB-d\fR \fIDir\fR] [\fB-f\fR
17 [\fB-f\fR \fIinactive\fR] [\fB-g\fR \fIgroup\fR] [\fB-G\fR \fIgroup\fR [, \fI
18 [\fB-K\fR \fIkey=value\fR] [\fB-m\fR [\fB-k\fR \fIskel_dir\fR]] [\fB-p\fR \
19 [\fB-P\fR \fIprofile\fR [, \fIprofile...\fR]] [\fB-R\fR \fIrole\fR [, \fIrole
20 \fB-s\fR \fIshell\fR] [\fB-u\fR \fIuid\fR] [\fB-o\fR]] \fIlogin\fR
21 .fi
22 .LP
23 .nf
24 \fBuseradd\fR \fB-D\fR [\fB-A\fR \fIauthorization\fR [, \fIauthorization...\fR]]
25 [\fB-b\fR \fIbase_dir\fR] [\fB-s\fR \fIshell\fR] [\fB-k\fR \fIskel_dir\fR]]
26 [\fB-f\fR \fIinactive\fR] [\fB-g\fR \fIgroup\fR] [\fB-G\fR \fIgroup\fR] [\fB-K\fR \fIkey=value\fR]
27 [\fB-P\fR \fIprofile\fR [, \fIprofile...\fR]] [\fB-R\fR \fIrole\fR [, \fIrole
28 .fi
29 .SH DESCRIPTION
30 .sp
31 .LP
32 \fBuseradd\fR adds a new user to the \fB/etc/passwd\fR and \fB/etc/shadow\fR
33 and \fB/etc/user_attr\fR files. The \fB-A\fR and \fB-P\fR options respectively
34 assign authorizations and profiles to the user. The \fB-R\fR option assigns
35 roles to a user. The \fB-p\fR option associates a project with a user. The
36 \fB-K\fR option adds a \fIkey=value\fR pair to \fB/etc/user_attr\fR for the
37 user. Multiple \fIkey=value\fR pairs may be added with multiple \fB-K\fR
38 options.
39 .sp
40 .LP
41 \fBuseradd\fR also creates supplementary group memberships for the user
42 (\fB-G\fR option) and creates the home directory (\fB-m\fR option) for the user
43 if requested. The new login remains locked until the \fBpasswd\fR(1) command is
44 executed.
45 .sp
46 .LP
47 Specifying \fBuseradd\fR \fB-D\fR with the \fB-s\fR, \fB-k\fR, \fB-g\fR,
48 \fB-b\fR, \fB-f\fR, \fB-e\fR, \fB-A\fR, \fB-P\fR, \fB-p\fR, \fB-R\fR, or
49 \fB-K\fR option (or any combination of these options) sets the default values
50 for the respective fields. See the \fB-D\fR option, below. Subsequent
51 \fBuseradd\fR commands without the \fB-D\fR option use these arguments.
52 .sp
53 .LP
54 The system file entries created with this command have a limit of 2048
55 characters per line. Specifying long arguments to several options can exceed
56 this limit.
57 .sp
58 .LP

```

```

60 \fBuseradd\fR requires that usernames be in the format described in
61 \fBpasswd\fR(4). A warning message is displayed if these restrictions are not
62 met. See \fBpasswd\fR(4) for the requirements for usernames.
63 .LP
64 To change the action of \fBuseradd\fR when the traditional login name
65 length limit of eight characters is exceeded, edit the file
66 \fB/etc/default/useradd\fR by removing the \fB#\fR (pound sign) before the
67 appropriate \fBEXCEED_TRAD=\fR entry, and adding it before the others.
68 .SH OPTIONS
69 .sp
70 .LP
71 The following options are supported:
72 .sp
73 .ne 2
74 .na
75 \fB\fB-A\fR \fIauthorization\fR\fR
76 .ad
77 .sp .6
78 .RS 4n
79 One or more comma separated authorizations defined in \fBauth_attr\fR(4). Only
80 a user or role who has \fBgrant\fR rights to the authorization can assign it to
81 an account.
82 .RE
83 .sp
84 .sp
85 .ne 2
86 .na
87 \fB\fB-b\fR \fIbase_dir\fR\fR
88 .ad
89 .sp .6
90 .RS 4n
91 The base directory for new login home directories (see the \fB-d\fR option
92 below. When a new user account is being created, \fBbase_dir\fR must already
93 exist unless the \fB-m\fR option or the \fB-d\fR option is also specified.
94 .RE
95 .sp
96 .sp
97 .ne 2
98 .na
99 \fB\fB-c\fR \fIcomment\fR\fR
100 .ad
101 .sp .6
102 .RS 4n
103 Any text string. It is generally a short description of the login, and is
104 currently used as the field for the user's full name. This information is
105 stored in the user's \fB/etc/passwd\fR entry.
106 .RE
107 .sp
108 .sp
109 .ne 2
110 .na
111 \fB\fB-d\fR \fIDir\fR\fR
112 .ad
113 .sp .6
114 .RS 4n
115 The home directory of the new user. It defaults to
116 \fBbase_dir\fR/\fIaccount_name\fR, where \fBbase_dir\fR is the base directory
117 for new login home directories and \fIaccount_name\fR is the new login name.
118 .RE
119 .sp
120 .sp
121 .ne 2
122 .na
123 \fB\fB-D\fR\fR
124 .ad
125 .sp .6

```

```

126 .RS 4n
127 Display the default values for \fBgroup\fR, \fBbase_dir\fR, \fBskel_dir\fR,
128 \fBshell\fR, \fBinactive\fR, \fBexpire\fR, \fBproj\fR, \fBprojname\fR and
129 \fBkey=value\fR pairs. When used with the \fB-g\fR, \fB-b\fR, \fB-f\fR,
130 \fB-e\fR, \fB-A\fR, \fB-P\fR, \fB-p\fR, \fB-R\fR, or \fB-K\fR options, the
131 \fB-D\fR option sets the default values for the specified fields. The default
132 values are:
133 .sp
134 .ne 2
135 .na
136 \fBgroup\fR
137 .ad
138 .sp .6
139 .RS 4n
140 \fBOther\fR (\fBGID\fR of 1)
141 .RE

143 .sp
144 .ne 2
145 .na
146 \fBbase_dir\fR
147 .ad
148 .sp .6
149 .RS 4n
150 \fBhome\fR
151 .RE

153 .sp
154 .ne 2
155 .na
156 \fBskel_dir\fR
157 .ad
158 .sp .6
159 .RS 4n
160 \fB/etc/skel\fR
161 .RE

163 .sp
164 .ne 2
165 .na
166 \fBshell\fR
167 .ad
168 .sp .6
169 .RS 4n
170 \fB/bin/sh\fR
171 .RE

173 .sp
174 .ne 2
175 .na
176 \fBinactive\fR
177 .ad
178 .sp .6
179 .RS 4n
180 \fB0\fR
181 .RE

183 .sp
184 .ne 2
185 .na
186 \fBexpire\fR
187 .ad
188 .sp .6
189 .RS 4n
190 null
191 .RE

```

```

193 .sp
194 .ne 2
195 .na
196 \fBauths\fR
197 .ad
198 .sp .6
199 .RS 4n
200 null
201 .RE

203 .sp
204 .ne 2
205 .na
206 \fBprofiles\fR
207 .ad
208 .sp .6
209 .RS 4n
210 null
211 .RE

213 .sp
214 .ne 2
215 .na
216 \fBproj\fR
217 .ad
218 .sp .6
219 .RS 4n
220 \fB3\fR
221 .RE

223 .sp
224 .ne 2
225 .na
226 \fBprojname\fR
227 .ad
228 .sp .6
229 .RS 4n
230 \fBdefault\fR
231 .RE

233 .sp
234 .ne 2
235 .na
236 \fBkey=value (pairs defined in \fBuser_attr\fR(4))\fR
237 .ad
238 .sp .6
239 .RS 4n
240 not present
241 .RE

243 .sp
244 .ne 2
245 .na
246 \fBroles\fR
247 .ad
248 .sp .6
249 .RS 4n
250 null
251 .RE

253 .RE

255 .sp
256 .ne 2
257 .na

```



```

258 \fB\fB-e\fR \fIexpire\fR\fR
259 .ad
260 .sp .6
261 .RS 4n
262 Specify the expiration date for a login. After this date, no user will be able
263 to access this login. The expire option argument is a date entered using one of
264 the date formats included in the template file \fB/etc/datemsk\fR. See
265 \fBgetdate\fR(3C).
266 .sp
267 If the date format that you choose includes spaces, it must be quoted. For
268 example, you can enter \fB10/6/90\fR or \fBOctober 6, 1990\fR. A null value
269 (\fB" \fR) defeats the status of the expired date. This option is useful for
270 creating temporary logins.
271 .RE

273 .sp
274 .ne 2
275 .na
276 \fB\fB-f\fR \fIinactive\fR\fR
277 .ad
278 .sp .6
279 .RS 4n
280 The maximum number of days allowed between uses of a login ID before that
281 \fBID\fR is declared invalid. Normal values are positive integers. A value of
282 \fB0\fR defeats the status.
283 .RE

285 .sp
286 .ne 2
287 .na
288 \fB\fB-g\fR \fIgroup\fR\fR
289 .ad
290 .sp .6
291 .RS 4n
292 An existing group's integer \fBID\fR or character-string name. Without the
293 \fB-D\fR option, it defines the new user's primary group membership and
294 defaults to the default group. You can reset this default value by invoking
295 \fBuseradd\fR \fB-D\fR \fB-g\fR \fIgroup\fR. GIDs 0-99 are reserved for
296 allocation by the Solaris Operating System.
297 .RE

299 .sp
300 .ne 2
301 .na
302 \fB\fB-G\fR \fIgroup\fR\fR
303 .ad
304 .sp .6
305 .RS 4n
306 An existing group's integer \fBID\fR or character-string name. It defines the
307 new user's supplementary group membership. Duplicates between \fIgroup\fR with
308 the \fB-g\fR and \fB-G\fR options are ignored. No more than \fBN_GROUPS_MAX\fR
309 groups can be specified. GIDs 0-99 are reserved for allocation by the Solaris
310 Operating System.
311 .RE

313 .sp
314 .ne 2
315 .na
316 \fB\fB-K\fR \fIkey=value\fR\fR
317 .ad
318 .sp .6
319 .RS 4n
320 A \fIkey=value\fR pair to add to the user's attributes. Multiple \fB-K\fR
321 options may be used to add multiple \fIkey=value\fR pairs. The generic \fB-K\fR
322 option with the appropriate key may be used instead of the specific implied key
323 options (\fB-A\fR, \fB-P\fR, \fB-R\fR, \fB-p\fR). See \fBuser_attr\fR(4) for a

```

```

324 list of valid \fIkey=value\fR pairs. The "type" key is not a valid key for this
325 option. Keys may not be repeated.
326 .RE

328 .sp
329 .ne 2
330 .na
331 \fB\fB-k\fR \fIskel_dir\fR\fR
332 .ad
333 .sp .6
334 .RS 4n
335 A directory that contains skeleton information (such as \fB&.profile\fR) that
336 can be copied into a new user's home directory. This directory must already
337 exist. The system provides the \fB/etc/skel\fR directory that can be used for
338 this purpose.
339 .RE

341 .sp
342 .ne 2
343 .na
344 \fB\fB-m\fR\fR
345 .ad
346 .sp .6
347 .RS 4n
348 Create the new user's home directory if it does not already exist. If the
349 directory already exists, it must have read, write, and execute permissions by
350 \fIgroup\fR, where \fIgroup\fR is the user's primary group.
351 .RE

353 .sp
354 .ne 2
355 .na
356 \fB\fB-o\fR\fR
357 .ad
358 .sp .6
359 .RS 4n
360 This option allows a \fBUID\fR to be duplicated (non-unique).
361 .RE

363 .sp
364 .ne 2
365 .na
366 \fB\fB-P\fR \fIprofile\fR\fR
367 .ad
368 .sp .6
369 .RS 4n
370 One or more comma-separated execution profiles defined in \fBprof_attr\fR(4).
371 .RE

373 .sp
374 .ne 2
375 .na
376 \fB\fB-p\fR \fIprojname\fR\fR
377 .ad
378 .sp .6
379 .RS 4n
380 Name of the project with which the added user is associated. See the
381 \fIprojname\fR field as defined in \fBproject\fR(4).
382 .RE

384 .sp
385 .ne 2
386 .na
387 \fB\fB-R\fR \fIrole\fR\fR
388 .ad
389 .sp .6

```

```

390 .RS 4n
391 One or more comma-separated execution profiles defined in \fBuser_attr\fR(4).
392 Roles cannot be assigned to other roles.
393 .RE

395 .sp
396 .ne 2
397 .na
398 \fB\fb-s\fR \fB\fbshell\fR
399 .ad
400 .sp .6
401 .RS 4n
402 Full pathname of the program used as the user's shell on login. It defaults to
403 an empty field causing the system to use \fB/bin/sh\fR as the default. The
404 value of \fB\fbshell\fR must be a valid executable file.
405 .RE

407 .sp
408 .ne 2
409 .na
410 \fB\fb-u\fR \fB\fbuid\fR
411 .ad
412 .sp .6
413 .RS 4n
414 The \fB\fbuid\fR of the new user. This \fB\fbuid\fR must be a non-negative decimal
415 integer below \fB\fbmaxuid\fR as defined in \fB<sys/param.h>\fR. The \fB\fbuid\fR
416 defaults to the next available (unique) number above the highest number
417 currently assigned. For example, if \fB\fbuid\fRs 100, 105, and 200 are assigned,
418 the next default \fB\fbuid\fR number will be 201. \fB\fbuid\fRs \fB\fb0\fR-\fB\fb99\fR are
419 reserved for allocation by the Solaris Operating System.
420 .RE

422 .SH FILES
423 .sp
424 .LP
425 \fB/etc/default/useradd\fR
426 .sp
427 .LP
428 \fB/etc/datemsk\fR
429 .sp
430 .LP
431 \fB/etc/passwd\fR
432 .sp
433 .LP
434 \fB/etc/shadow\fR
435 .sp
436 .LP
437 \fB/etc/group\fR
438 .sp
439 .LP
440 \fB/etc/skel\fR
441 .sp
442 .LP
443 \fB/usr/include/limits.h\fR
444 .sp
445 .LP
446 \fB/etc/user_attr\fR
447 .SH ATTRIBUTES
448 .sp
449 .LP
450 See \fB\fbattributes\fR(5) for descriptions of the following attributes:
451 .sp

453 .sp
454 .TS
455 box;

```

```

456 c | c
457 l | l .
458 ATTRIBUTE TYPE ATTRIBUTE VALUE
459 -
460 Interface Stability Committed
461 .TE

463 .SH SEE ALSO
464 .sp
465 .LP
466 \fB\fbpasswd\fR(1), \fB\fbprofiles\fR(1), \fB\fbroles\fR(1), \fB\fbusers\fR(1B),
467 \fB\fbgroupadd\fR(1M), \fB\fbgroupdel\fR(1M), \fB\fbgroupmod\fR(1M), \fB\fbgrpck\fR(1M),
468 \fB\fblogins\fR(1M), \fB\fbpwck\fR(1M), \fB\fbuserdel\fR(1M), \fB\fbusermod\fR(1M),
469 \fB\fbgetdate\fR(3C), \fB\fbauth_attr\fR(4), \fB\fbpasswd\fR(4), \fB\fbprof_attr\fR(4),
470 \fB\fbproject\fR(4), \fB\fbuser_attr\fR(4), \fB\fbattributes\fR(5)
471 .SH DIAGNOSTICS
472 .sp
473 .LP
474 In case of an error, \fBuseradd\fR prints an error message and exits with a
475 non-zero status.
476 .sp
477 .LP
478 The following indicates that \fBlogin\fR specified is already in use:
479 .sp
480 .in +2
481 .nf
482 UX: useradd: ERROR: login is already in use. Choose another.
483 .fi
484 .in -2
485 .sp

487 .sp
488 .LP
489 The following indicates that the \fB\fbuid\fR specified with the \fB\fb-u\fR option
490 is not unique:
491 .sp
492 .in +2
493 .nf
494 UX: useradd: ERROR: uid \fB\fbuid\fR is already in use. Choose another.
495 .fi
496 .in -2
497 .sp

499 .sp
500 .LP
501 The following indicates that the \fB\fbgroup\fR specified with the \fB\fb-g\fR option
502 is already in use:
503 .sp
504 .in +2
505 .nf
506 UX: useradd: ERROR: group \fB\fbgroup\fR does not exist. Choose another.
507 .fi
508 .in -2
509 .sp

511 .sp
512 .LP
513 The following indicates that the \fB\fbuid\fR specified with the \fB\fb-u\fR option
514 is in the range of reserved \fB\fbuid\fRs (from \fB\fb0\fR-\fB\fb99\fR):
515 .sp
516 .in +2
517 .nf
518 UX: useradd: WARNING: uid \fB\fbuid\fR is reserved.
519 .fi
520 .in -2
521 .sp

```

```
523 .sp
524 .LP
525 The following indicates that the \fIuid\fR specified with the \fB-u\fR option
526 exceeds \fBMAXUID\fR as defined in \fB<sys/param.h>\fR:
527 .sp
528 .in +2
529 .nf
530 UX: useradd: ERROR: uid \fIuid\fR is too big. Choose another.
531 .fi
532 .in -2
533 .sp

535 .sp
536 .LP
537 The following indicates that the \fB/etc/passwd\fR or \fB/etc/shadow\fR files
538 do not exist:
539 .sp
540 .in +2
541 .nf
542 UX: useradd: ERROR: Cannot update system files - login cannot be created.
543 .fi
544 .in -2
545 .sp

547 .SH NOTES
548 .sp
549 .LP
550 The \fBuseradd\fR utility adds definitions to only the local \fB/etc/group\fR,
551 \fB/etc/passwd\fR, \fB/etc/passwd\fR, \fB/etc/shadow\fR, \fB/etc/project\fR, and
552 \fB/etc/user_attr\fR files. If a network name service such as \fBNIS\fR or
553 \fBNIS+\fR is being used to supplement the local \fB/etc/passwd\fR file with
554 additional entries, \fBuseradd\fR cannot change information supplied by the
555 network name service. However \fBuseradd\fR will verify the uniqueness of the
556 user name (or role) and user id and the existence of any group names specified
557 against the external name service.
```

```

*****
7698 Sat Jun 1 10:02:51 2013
new/usr/src/man/man3c/getlogin.3c
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
1 \" te
2 .\" Copyright (c) 2013 Gary Mills
3 .\" Copyright (c) 2004 Sun Microsystems, Inc. All Rights Reserved.
4 .\" Copyright 1989 AT&T
5 .\" Portions Copyright (c) 1992, X/Open Company Limited. All Rights Reserved.
6 .\" Sun Microsystems, Inc. gratefully acknowledges The Open Group for permission
7 .\" http://www.opengroup.org/bookstore/.
8 .\" The Institute of Electrical and Electronics Engineers and The Open Group, ha
9 .\" This notice shall appear on any product containing this material.
10 .\" The contents of this file are subject to the terms of the Common Development
11 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
12 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
13 .TH GETLOGIN 3C \"May 25, 2013\"
14 .SH NAME
15 getlogin, getlogin_r \- get login name
16 .SH SYNOPSIS
17 .LP
18 .nf
19 #include <unistd.h>

21 \fBchar *\fR\fBgetlogin\fR(\fBvoid\fR);
22 .fi

24 .LP
25 .nf
26 \fBchar *\fR\fBgetlogin_r\fR(\fBchar *\fR\fBname\fR, \fBint\fR \fBnamelen\fR);
27 .fi

29 .SS \"Standard conforming\"
30 .LP
31 .nf
32 cc [ \fBiflag \fR... ] \fBifile\fR... \fB-D_POSIX_PTHREAD_SEMANTICS\fR [ \fBilibrar

34 \fBint\fR \fBgetlogin_r\fR(\fBchar *\fR\fBname\fR, \fBsize_t\fR \fBnamesize\fR);
35 .fi

37 .SH DESCRIPTION
38 .sp
39 .LP
40 The \fBgetlogin()\fR function returns a pointer to the login name as found in
41 \fB/var/adm/utmpx\fR. It can be used in conjunction with \fBgetpwnam\fR(3C) to
42 locate the correct password file entry when the same user \fBUID\fR is shared by
43 several login names.
44 .sp
45 .LP
46 The login name plus the terminating null byte can be up to 33 characters
47 in length.
48 Newly-compiled programs should use the \fBLOGIN_NAME_MAX\fR symbol,
49 defined in <\fBlimits.h\fR>, to size the buffer.
50 Older programs that call \fBgetlogin()\fR expect only the legacy
51 9-character length.
52 These automatically link to a version of the \fBgetlogin()\fR functions that
53 truncates longer login names.
54 It's also possible to compile new programs that link to truncating versions
55 of these functions by defining \fB_USE_LEGACY_LOGNAME_\fR in the
56 compile environment.
57 .sp
58 .LP
59 Some older programs will correctly handle long login names returned

```

```

60 by the \fBgetlogin()\fR function.
61 For this case, the user compatibility library
62 \fB/usr/lib/getloginx.so.1\fR redirects to a version of the \fBgetlogin()\fR
63 function that returns the long name.
64 This library should be added to such an application
65 at runtime using \fBBLD_PRELOAD\fR.
66 .sp
67 .LP
68 If \fBgetlogin()\fR is called within a process that is not attached to a
69 terminal, it returns a null pointer. The correct procedure for determining the
70 login name is to call \fBcuserid\fR(3C), or to call \fBgetlogin()\fR and if it
71 fails to call \fBgetpwuid\fR(3C).
72 .sp
73 .LP
74 The \fBgetlogin_r()\fR function has the same functionality as \fBgetlogin()\fR
75 except that the caller must supply a buffer \fBname\fR with length
76 \fBnamelen\fR to store the result. The \fBname\fR buffer must be at least
77 \fBLOGIN_NAME_MAX\fR bytes in size (defined in <\fBlimits.h\fR>). The
78 \fB_POSIX_LOGIN_NAME_MAX\fR bytes in size (defined in <\fBlimits.h\fR>). The
79 POSIX version (see \fBstandards\fR(5)) of \fBgetlogin_r()\fR takes a
80 \fBnamesize\fR parameter of type \fBsize_t\fR.
81 .SH RETURN VALUES
82 .sp
83 .LP
84 Upon successful completion, \fBgetlogin()\fR returns a pointer to the login
85 name or a null pointer if the user's login name cannot be found. Otherwise it
86 returns a null pointer and sets \fBerrno\fR to indicate the error.
87 .sp
88 .LP
89 The standard-conforming \fBgetlogin_r()\fR returns \fB0\fR if successful, or
90 the error number upon failure.
91 .SH ERRORS
92 .sp
93 .LP
94 The \fBgetlogin_r()\fR function will fail if:
95 .ne 2
96 .na
97 \fB\BERANGE\fR
98 .ad
99 .RS 10n
100 The size of the buffer is smaller than the result to be returned.
101 .RE

103 .sp
104 .ne 2
105 .na
106 \fB\BEINVAL\fR
107 .ad
108 .RS 10n
109 And entry for the current user was not found in the \fB/var/adm/utmpx\fR file.
110 .RE

112 .sp
113 .LP
114 The \fBgetlogin()\fR and \fBgetlogin_r()\fR functions may fail if:
115 .sp
116 .ne 2
117 .na
118 \fB\BEMFILE\fR
119 .ad
120 .RS 10n
121 There are {\fBOPEN_MAX\fR} file descriptors currently open in the calling
122 process.
123 .RE

```

```

125 .sp
126 .ne 2
127 .na
128 \fB\fBENFILE\fR\fR
129 .ad
130 .RS 10n
131 The maximum allowable number of files is currently open in the system.
132 .RE

134 .sp
135 .ne 2
136 .na
137 \fB\fBENXIO\fR\fR
138 .ad
139 .RS 10n
140 The calling process has no controlling terminal.
141 .RE

143 .sp
144 .LP
145 The \fBgetlogin_r()\fR function may fail if:
146 .sp
147 .ne 2
148 .na
149 \fB\fBERANGE\fR\fR
150 .ad
151 .RS 10n
152 The size of the buffer is smaller than the result to be returned.
153 .RE

155 .SH USAGE
156 .sp
157 .LP
158 The return value of \fBgetlogin()\fR points to thread-specific data whose
159 content is overwritten on each call by the same thread.
160 .sp
161 .LP
162 Three names associated with the current process can be determined:
163 \fBgetpwnam()\fR, \fBgetuid()\fR, \fBgetlogin()\fR returns the name associated with the
164 effective user ID of the process; \fBgetlogin()\fR returns the name associated
165 with the current login activity; and \fBgetpwnam()\fR, \fBgetuid()\fR, \fBgetlogin()\fR
166 returns the name associated with the real user ID of the process.
167 .SH FILES
168 .sp
169 .ne 2
170 .na
171 \fB/var/adm/utmpx\fR
172 .ad
173 .RS 18n
174 user access and administration information
175 .RE

177 .sp
178 .ne 2
179 .na
180 \fB/usr/lib/getloginx.so.1\fR
181 .ad
182 .RS 18n
183 A compatibility library that returns long login names to older applications.
184 .RE

186 .sp
187 .ne 2
188 .na
189 \fB/usr/lib/64/getloginx.so.1\fR
190 .ad

```

```

191 .RS 18n
192 A 64-bit compatibility library to return long login names.
193 .RE

195 .SH ATTRIBUTES
196 .sp
197 .LP
198 See \fBattributes\fR(5) for descriptions of the following attributes:
199 .sp

201 .sp
202 .TS
203 box;
204 c | c
205 l | l .
206 ATTRIBUTE TYPE ATTRIBUTE VALUE
207 _
208 Interface Stability Standard
209 _
210 MT-Level See below.
211 .TE

213 .SH SEE ALSO
214 .sp
215 .LP
216 \fBgetuid()\fR, \fBgetuid()\fR, \fBgetuid()\fR, \fBgetgrnam()\fR,
217 \fBgetpwnam()\fR, \fBgetpwnam()\fR, \fBgetpwnam()\fR, \fBgetpwnam()\fR, \fBgetpwnam()\fR,
218 \fBstandards\fR(5)
219 .SH NOTES
220 .sp
221 .LP
222 When compiling multithreaded programs, see \fBintro\fR(3).
223 .sp
224 .LP
225 The \fBgetlogin()\fR function is safe to use in multithreaded applications, but
226 is discouraged. The \fBgetlogin_r()\fR function should be used instead.
227 .sp
228 .LP
229 Solaris 2.4 and earlier releases provided a \fBgetlogin_r()\fR as specified in
230 POSIX.1c Draft 6. The final POSIX.1c standard changed the interface as
231 described above. Support for the Draft 6 interface is provided for
232 compatibility only and may not be supported in future releases. New
233 applications and libraries should use the standard-conforming interface.

```



126	_SC_EXPR_NEST_MAX	EXPR_NEST_MAX	Max number of parentheses by expr
127			Supports File Synchronization
128	_SC_FSYNC	_POSIX_FSYNC	Max size of group entry buffer
129			Max size of password entry buffer
130	_SC_GETGR_R_SIZE_MAX		Maximum length of a host name (excluding terminating null)
131			Max number of iovec structures available to one process for use with readv() and writev()
132	_SC_GETPW_R_SIZE_MAX		Job control supported?
133			Max length of input line
134	_SC_HOST_NAME_MAX	_POSIX_HOST_NAME_MAX	<b>Max length of login</b>
135			Max length of login name
136	_SC_IOV_MAX	IOV_MAX	Supports Memory Mapped Files
137			Max pid value
138			Supports Process Memory Locking
139			Supports Range Memory Locking
140			Supports Memory Protection
141	_SC_JOB_CONTROL	_POSIX_JOB_CONTROL	Supports Message Passing
142			Supports Monotonic Clock option
143	_SC_LINE_MAX	LINE_MAX	Max number of open message queues a process can hold
144			Max number of message priorities supported
145	_SC_LOGIN_NAME_MAX	LOGIN_NAME_MAX	Max simultaneous groups to which one can belong
146	_SC_LOGIN_NAME_MAX	LOGIN_NAME_MAX	Number of processors configured
147			Max number of processors supported by platform
148	_SC_LOGNAME_MAX	LOGNAME_MAX	Number of processors online
149	_SC_LOGNAME_MAX	LOGNAME_MAX	Max open files per process
150	_SC_MAPPED_FILES	_POSIX_MAPPED_FILES	System memory page size
151			Same as _SC_PAGESIZE
152	_SC_MAXPID		Max number of significant bytes in a password
153	_SC_MEMLOCK	_POSIX_MEMLOCK	Total number of pages of physical memory in system
154			Supports Prioritized I/O
155	_SC_MEMLOCK_RANGE	_POSIX_MEMLOCK_RANGE	
156	_SC_MEMORY_PROTECTION	_POSIX_MEMORY_PROTECTION	
157	_SC_MESSAGE_PASSING	_POSIX_MESSAGE_PASSING	
158	_SC_MESSAGE_PASSING	_POSIX_MESSAGE_PASSING	
159	_SC_MONOTONIC_CLOCK	_POSIX_MONOTONIC_CLOCK	
160			
161	_SC_MQ_OPEN_MAX	MQ_OPEN_MAX	
162			
163	_SC_MQ_OPEN_MAX	MQ_OPEN_MAX	
164			
165	_SC_MQ_PRIO_MAX	MQ_PRIO_MAX	
166			
167			
168	_SC_NGROUPS_MAX	NGROUPS_MAX	
169			
170			
171	_SC_NPROCESSORS_CONF		
172			
173	_SC_NPROCESSORS_MAX		
174			
175			
176	_SC_NPROCESSORS_ONLN		
177			
178	_SC_OPEN_MAX	OPEN_MAX	
179			
180	_SC_PAGESIZE	PAGESIZE	
181			
182	_SC_PAGE_SIZE	PAGESIZE	
183	_SC_PASS_MAX	PASS_MAX	
184			
185			
186	_SC_PHYS_PAGES		
187			
188			
189	_SC_PRIORITIZED_IO	_POSIX_PRIORITIZED_IO	
190			

191	_SC_PRIORITY_SCHEDULING	_POSIX_PRIORITY_SCHEDULING	Supports Process Scheduling
192			Supports Raw Sockets option
193	_SC_RAW_SOCKETS	_POSIX_RAW_SOCKETS	Max number of repeated occurrences of a regular expression permitted when using interval notation \e{m,n\e}
194			Supports IPV6 option
195	_SC_RE_DUP_MAX	RE_DUP_MAX	Supports Realtime Signals
196			Supports Regular Expression Handling option
197			Max number of realtime signals reserved for application use
198			Supported?
199			Max number of POSIX semaphores a process can have
200			Max value a POSIX semaphore can have
201	_SC_READER_WRITER_LOCKS	_POSIX_READER_WRITER_LOCKS	Supports Semaphores
202	_SC_REALTIME_SIGNALS	_POSIX_REALTIME_SIGNALS	Supports Shared Memory Objects
203			Supports POSIX shell
204	_SC_REGEX	_POSIX_REGEX	Max number of queued signals that a process can send and have pending at receiver(s) at a time
205			Supports Spawn option
206			Supports Spin Locks option
207	_SC_RTSIG_MAX	RTSIG_MAX	Default stack protection
208			Number of streams one process can have open at a time
209			Max number of symbolic links that can be reliably traversed in the resolution of a pathname in the absence of a loop
210			Supports Synchronized I/O
211	_SC_SAVED_IDS	_POSIX_SAVED_IDS	Supports Thread Stack Address Attribute option
212			Supports Thread Stack Size Attribute option
213			Number attempts made to destroy thread-specific data on thread exit
214	_SC_SEM_NSEMS_MAX	SEM_NSEMS_MAX	Max number of data keys per process
215			
216			
217	_SC_SEM_VALUE_MAX	SEM_VALUE_MAX	
218			
219	_SC_SEMAPHORES	_POSIX_SEMAPHORES	
220	_SC_SHARED_MEMORY_OBJECTS	_POSIX_SHARED_MEMORY_OBJECTS	
221	_SC_SHELL	_POSIX_SHELL	
222	_SC_SHELL	_POSIX_SHELL	
223	_SC_SIGQUEUE_MAX	SIGQUEUE_MAX	
224			
225			
226			
227			
228			
229	_SC_SPAWN	_POSIX_SPAWN	
230	_SC_SPIN_LOCKS	_POSIX_SPIN_LOCKS	
231			
232	_SC_STACK_PROT		
233			
234	_SC_STREAM_MAX	STREAM_MAX	
235			
236			
237	_SC_SYMLINK_MAX	_POSIX_SYMLINK_MAX	
238			
239			
240			
241			
242			
243	_SC_SYNCHRONIZED_IO	_POSIX_SYNCHRONIZED_IO	
244			
245	_SC_THREAD_ATTR_STACKADDR	_POSIX_THREAD_ATTR_STACKADDR	
246	_SC_THREAD_ATTR_STACKADDR	_POSIX_THREAD_ATTR_STACKADDR	
247			
248	_SC_THREAD_ATTR_STACKSIZE	_POSIX_THREAD_ATTR_STACKSIZE	
249	_SC_THREAD_ATTR_STACKSIZE	_POSIX_THREAD_ATTR_STACKSIZE	
250			
251	_SC_THREAD_DESTRUCTOR_ITERATIONS	PTHREAD_DESTRUCTOR_ITERATIONS	
252	_SC_THREAD_DESTRUCTOR_ITERATIONS	PTHREAD_DESTRUCTOR_ITERATIONS	
253			
254			
255	_SC_THREAD_KEYS_MAX	PTHREAD_KEYS_MAX	
256			

```

257 _SC_THREAD_PRIO_      _POSIX_THREAD_PRIO_      Supports Priority
258     INHERIT            INHERIT                    Inheritance option
259 _SC_THREAD_PRIO_      _POSIX_THREAD_PRIO_      Supports Priority
260     PROTECT            PROTECT                    Protection option
261 _SC_THREAD_PRIORITY_  _POSIX_THREAD_PRIORITY_  Supports Thread
262     SCHEDULING         SCHEDULING                 Execution
263                                     Scheduling option
264 _SC_THREAD_PROCESS_   _POSIX_THREAD_PROCESS_   Supports
265     SHARED              SHARED                     Process-Shared
266                                     Synchronization
267                                     option
268 _SC_THREAD_SAFE_      _POSIX_THREAD_SAFE_      Supports Thread-Safe
269     FUNCTIONS           FUNCTIONS                   Functions option
270 _SC_THREAD_STACK_MIN  PTHREAD_STACK_MIN        Min byte size of
271                                     thread stack storage
272 _SC_THREAD_THREADS_MAX PTHREAD_THREADS_MAX      Max number of
273                                     threads per process
274 _SC_THREADS            _POSIX_THREADS            Supports Threads
275                                     option
276 _SC_TIMEOULTS         _POSIX_TIMEOULTS         Supports Timeouts
277                                     option
278 _SC_TIMER_MAX         TIMER_MAX                 Max number of timer
279                                     per process
280                                     supported
281 _SC_TIMERS            _POSIX_TIMERS             Supports Timers
282 _SC_TTY_NAME_MAX     TTYNAME_MAX              Max length of tty
283                                     device name
284 _SC_TZNAME_MAX        TZNAME_MAX                Max number of bytes
285                                     supported for name
286                                     of a time zone
287 _SC_V6_ILP32_OFF32    _POSIX_V6_ILP32_OFF32    Supports X/Open
288                                     ILP32 w/32-bit
289                                     offset build
290                                     environment
291 _SC_V6_ILP32_OFFBIG   _POSIX_V6_ILP32_OFFBIG   Supports X/Open
292                                     ILP32 w/64-bit
293                                     offset build
294                                     environment
295 _SC_V6_LP64_OFF64     _POSIX_V6_LP64_OFF64     Supports X/Open
296                                     LP64 w/64-bit
297                                     offset build
298                                     environment
299 _SC_V6_LPBIG_OFFBIG   _POSIX_V6_LPBIG_OFFBIG   Same as
300                                     _SC_V6_LP64_OFF64
301 _SC_VERSION           _POSIX_VERSION            POSIX.1 version
302                                     supported
303 _SC_XBS5_ILP32_OFF32  _XBS5_ILP32_OFF32        Indicates support
304                                     for X/Open ILP32
305                                     w/32-bit offset
306                                     build environment
307 _SC_XBS5_ILP32_OFFBIG _XBS5_ILP32_OFFBIG        Indicates support
308                                     for X/Open ILP32
309                                     w/64-bit offset
310                                     build environment
311 _SC_XBS5_LP64_OFF64   _XBS5_LP64_OFF64         Indicates support of
312                                     X/Open LP64,
313                                     64-bit offset
314                                     build environment
315 _SC_XBS5_LPBIG_OFFBIG _XBS5_LP64_OFF64         Same as
316                                     _SC_XBS5_LP64_OFF64
317 _SC_XOPEN_CRYPT       _XOPEN_CRYPT              Supports X/Open
318                                     Encryption Feature
319                                     Group
320 _SC_XOPEN_ENH_I18N    _XOPEN_ENH_I18N          Supports X/Open
321                                     Enhanced
322                                     Internationalization

```

```

323                                     Feature Group
324 _SC_XOPEN_LEGACY      _XOPEN_LEGACY             Supports X/Open
325                                     Legacy Feature Group
326 _SC_XOPEN_REALTIME    _XOPEN_REALTIME           Supports X/Open
327                                     POSIX Realtime
328                                     Feature Group
329 _SC_XOPEN_REALTIME_   _XOPEN_REALTIME_THREADS  Supports X/Open
330     THREADS              POSIX Reatime
331                                     Threads Feature
332                                     Group
333 _SC_XOPEN_SHM          _XOPEN_SHM                Supports X/Open
334                                     Shared Memory
335                                     Feature Group
336 _SC_XOPEN_STREAMS     _POSIX_XOPEN_STREAMS     Supports XSI Streams
337                                     option group
338 _SC_XOPEN_UNIX        _XOPEN_UNIX               Supports X/Open CAE
339                                     Specification,
340                                     August 1994, System
341                                     Interfaces and
342                                     Headers, Issue 4,
343                                     Version 2
344 _SC_XOPEN_VERSION     _XOPEN_VERSION            Integer value
345                                     indicates version of
346                                     X/Open Portability
347                                     Guide to which
348                                     implementation
349                                     conforms
350 _SC_XOPEN_XCU_VERSION _XOPEN_XCU_VERSION        Integer value
351                                     indicates version of
352                                     XCU specification to
353                                     which implementation
354                                     conforms
355 .fi
356 .in -2
357 .sp
358
359 .sp
360 .LP
361 The following options are not supported and return \ (mil:
362 .sp
363
364 .sp
365 .TS
366 l l
367 l l .
368 \fb_SC_2_PBS\fr \fb_POSIX2_PBS\fr
369 \fb_SC_2_PBS_ACCOUNTING\fr \fb_POSIX2_PBS_ACCOUNTING\fr
370 \fb_SC_2_PBS_CHECKPOINT\fr \fb_POSIX2_PBS_CHECKPOINT\fr
371 \fb_SC_2_PBS_LOCATE\fr \fb_POSIX2_PBS_LOCATE\fr
372 \fb_SC_2_PBS_MESSAGE\fr \fb_POSIX2_PBS_MESSAGE\fr
373 \fb_SC_2_PBS_TRACK\fr \fb_POSIX2_PBS_TRACK\fr
374 \fb_SC_THREAD_ADVISORY_INFO\fr \fb_POSIX_THREAD_ADVISORY_INFO\fr
375 \fb_SC_CPUTIME\fr \fb_POSIX_CPUTIME\fr
376 \fb_SC_SPORADIC_SERVER\fr \fb_POSIX_SPORADIC_SERVER\fr
377 \fb_SC_SS_REPL_MAX\fr \fb_POSIX_SS_REPL_MAX\fr
378 \fb_SC_THREAD_CPUTIME\fr \fb_POSIX_THREAD_CPUTIME\fr
379 \fb_SC_THREAD_SPORADIC_SERVER\fr \fb_POSIX_THREAD_SPORADIC_SERVER\fr
380 \fb_SC_TRACE\fr \fb_POSIX_TRACE\fr
381 \fb_SC_TRACE_EVENT_FILTER\fr \fb_POSIX_TRACE_EVENT_FILTER\fr
382 \fb_SC_TRACE_EVENT_NAME_MAX\fr \fb_POSIX_TRACE_EVENT_NAME_MAX\fr
383 \fb_SC_TRACE_INHERIT\fr \fb_POSIX_TRACE_INHERIT\fr
384 \fb_SC_TRACE_LOG\fr \fb_POSIX_TRACE_LOG\fr
385 \fb_SC_TRACE_NAME_MAX\fr \fb_POSIX_TRACE_NAME_MAX\fr
386 \fb_SC_TRACE_SYS_MAX\fr \fb_POSIX_TRACE_SYS_MAX\fr
387 \fb_SC_TRACE_USER_EVENT_MAX\fr \fb_POSIX_TRACE_USER_EVENT_MAX\fr
388 \fb_SC_TYPED_MEMORY_OBJECTS\fr \fb_POSIX_TYPED_MEMORY_OBJECTS\fr

```



```

389 .TE

391 .SH RETURN VALUES
392 .sp
393 .LP
394 Upon successful completion, \fBsysconf()\fR returns the current variable value
395 on the system. The value returned will not be more restrictive than the
396 corresponding value described to the application when it was compiled with the
397 implementation's <\fBlimits.h\fR>, <\fBunistd.h\fR> or <\fBtime.h\fR>. With
398 only a few obvious exceptions such as \fB_SC_AVPHYS_PAGES\fR and
399 \fB_SC_NPROCESSORS_ONLN\fR, the value will not change during the lifetime of
400 the calling process.
401 .sp
402 .LP
403 If \fIname\fR is an invalid value, \fBsysconf()\fR returns \fB\{mil\fR and sets
404 \fBerrno\fR to indicate the error. If the variable corresponding to \fIname\fR
405 is associated with functionality that is not supported by the system,
406 \fBsysconf()\fR returns \fB\{mil\fR without changing the value of \fBerrno\fR.
407 .sp
408 .LP
409 Calling \fBsysconf()\fR with the following returns \fB\{mil\fR without setting
410 \fBerrno\fR, because no maximum limit can be determined. The system supports at
411 least the minimum values and can support higher values depending upon system
412 resources.
413 .sp
414 .in +2
415 .nf
416 Variable                Minimum supported value
417 _SC_AIO_MAX              _POSIX_AIO_MAX
418 _SC_ATEXIT_MAX          32
419 _SC_MQ_OPEN_MAX         32
420 _SC_THREAD_THREADS_MAX  _POSIX_THREAD_THREADS_MAX
421 _SC_THREAD_KEYS_MAX     _POSIX_THREAD_KEYS_MAX
422 _SC_THREAD_DESTRUCTOR_ITERATIONS  _POSIX_THREAD_DESTRUCTOR_ITERATIONS
423 .fi
424 .in -2

426 .sp
427 .LP
428 The following SPARC and x86 platform variables return \fBEINVAL\fR:
429 .sp
430 .in +2
431 .nf
432 _SC_COHER_BLKSIZE      _SC_DCACHE_ASSOC
433 _SC_DCACHE_BLKSIZE    _SC_DCACHE_LINESZ
434 _SC_DCACHE_SZ          _SC_DCACHE_TBLKSZ
435 _SC_ICACHE_ASSOC      _SC_ICACHE_BLKSIZE
436 _SC_ICACHE_LINESZ    _SC_ICACHE_SZ
437 _SC_SPLIT_CACHE
438 .fi
439 .in -2

441 .SH ERRORS
442 .sp
443 .LP
444 The \fBsysconf()\fR function will fail if:
445 .sp
446 .ne 2
447 .na
448 \fBEINVAL\fR
449 .ad
450 .RS 10n
451 The value of the \fIname\fR argument is invalid.
452 .RE

454 .SH ATTRIBUTES

```

```

455 .sp
456 .LP
457 See \fBattributes\fR(5) for descriptions of the following attributes:
458 .sp

460 .sp
461 .TS
462 box;
463 c | c
464 l | l .
465 ATTRIBUTE TYPE    ATTRIBUTE VALUE
466 _
467 Architecture      SPARC and x86
468 _
469 Interface Stability    Committed
470 _
471 MT-Level           MT-Safe, Async-Signal-Safe
472 _
473 Standard           See \fBstandards\fR(5).
474 .TE

476 .SH SEE ALSO
477 .sp
478 .LP
479 \fBpooladm\fR(1M), \fBzoneadm\fR(1M), \fBfpathconf\fR(2), \fBseteuid\fR(2),
480 \fBsetrlimit\fR(2), \fBconfstr\fR(3C), \fBattributes\fR(5), \fBstandards\fR(5)
481 .SH NOTES
482 .sp
483 .LP
484 A call to \fBsetrlimit()\fR can cause the value of \fBOPEN_MAX\fR to change.
485 .sp
486 .LP
487 Multiplying \fBsysconf\fR(\fB_SC_PHYS_PAGES\fR) or
488 \fBsysconf\fR(\fB_SC_AVPHYS_PAGES\fR) by \fBsysconf\fR(\fB_SC_PAGESIZE\fR) to
489 determine memory amount in bytes can exceed the maximum values representable in
490 a 32-bit signed or unsigned integer.
491 .sp
492 .LP
493 The value of \fBCLK_TCK\fR can be variable and it should not be assumed that
494 \fBCLK_TCK\fR is a compile-time constant.
495 .sp
496 .LP
497 If the caller is in a non-global zone and the pools facility is active,
498 \fBsysconf\fR(\fB_SC_NPROCESSORS_CONF\fR) and
499 \fBsysconf\fR(\fB_SC_NPROCESSORS_ONLN\fR) return the number of processors in
500 the processor set of the pool to which the zone is bound.

```

```

*****
 9883 Sat Jun 1 10:02:51 2013
new/usr/src/man/man4/passwd.4
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
 1 \" te
 2 .\" Copyright (c) 2013 Gary Mills
 3 .\" Copyright (c) 2004, Sun Microsystems, Inc. All Rights Reserved.
 4 .\" Copyright 1989 AT&T
 5 .\" The contents of this file are subject to the terms of the Common Development
 6 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
 7 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
 8 .TH PASSWD 4 \"Apr 16, 2013\"
 9 .TH PASSWD 4 \"Jul 28, 2004\"
10 .SH NAME
11 passwd \- password file
12 .SH SYNOPSIS
13 .LP
14 \fB/etc/passwd\fR
15 .fi

17 .SH DESCRIPTION
18 .sp
19 .LP
20 The file \fB/etc/passwd\fR is a local source of information about users'
21 accounts. The password file can be used in conjunction with other naming
22 sources, such as the \fBNIS\fR maps \fBpasswd.byname\fR and \fBpasswd.bygid\fR,
23 data from the \fBNIS+\fR \fBpasswd\fR table, or password data stored on an LDAP
24 server. Programs use the \fBgetpwnam\fR(3C) routines to access this
25 information.
26 .sp
27 .LP
28 Each \fBpasswd\fR entry is a single line of the form:
29 .sp
30 .in +2
31 .nf
32 \fIusername\fR:\fB:\fR:\fIpassword\fR:\fB:\fR:\fIuid\fR:\fB:\fR
33 \fIgid\fR:\fB:\fR:\fIgcospwname\fR:\fB:\fR:\fIhome-dir\fR:\fB:\fR
34 \fIlogin-shell\fR
35 .fi
36 .in -2
37 .sp

39 .sp
40 .LP
41 where
42 .sp
43 .ne 2
44 .na
45 \fB\fIusername\fR\fR
46 .ad
47 .RS 15n
48 is the user's login name.
49 .sp
50 The login (\fBlogin\fR) and role (\fBrole\fR) fields accept a string of no more
51 than 32 bytes consisting of characters from the set of alphabetic
52 than eight bytes consisting of characters from the set of alphabetic
53 characters, numeric characters, period (\fB.&\fR), underscore (\fB_\fR), and
54 hyphen (\fB-\fR). The first character should be alphabetic and the field should
55 contain at least one lower case alphabetic character. A warning message is
56 displayed if these restrictions are not met.
57 .sp
58 The \fBlogin\fR and \fBrole\fR fields must contain at least one character and
59 must not contain a colon (\fB:\fR) or a newline (\fB\n\fR).

```

```

59 .RE

61 .sp
62 .ne 2
63 .na
64 \fB\fIpassword\fR\fR
65 .ad
66 .RS 15n
67 is an empty field. The encrypted password for the user is in the corresponding
68 entry in the \fB/etc/shadow\fR file. \fBpwconv\fR(1M) relies on a special value
69 of '\fBx\fR' in the password field of \fB/etc/passwd\fR. If this value
70 of '\fBx\fR' exists in the password field of \fB/etc/passwd\fR, this indicates
71 that the password for the user is already in \fB/etc/shadow\fR and should not
72 be modified.
73 .RE

75 .sp
76 .ne 2
77 .na
78 \fB\fIuid\fR\fR
79 .ad
80 .RS 15n
81 is the user's unique numerical \fBID\fR for the system.
82 .RE

84 .sp
85 .ne 2
86 .na
87 \fB\fIgid\fR\fR
88 .ad
89 .RS 15n
90 is the unique numerical \fBID\fR of the group that the user belongs to.
91 .RE

93 .sp
94 .ne 2
95 .na
96 \fB\fIgcospwname\fR\fR
97 .ad
98 .RS 15n
99 is the user's real name, along with information to pass along in a mail-message
100 heading. (It is called the gcospwname-field for historical reasons.) An '\fB&\fR'
101 (ampersand) in this field stands for the login name (in cases where the login
102 name appears in a user's real name).
103 .RE

105 .sp
106 .ne 2
107 .na
108 \fB\fIhome-dir\fR\fR
109 .ad
110 .RS 15n
111 is the pathname to the directory in which the user is initially positioned upon
112 logging in.
113 .RE

115 .sp
116 .ne 2
117 .na
118 \fB\fIlogin-shell\fR\fR
119 .ad
120 .RS 15n
121 is the user's initial shell program. If this field is empty, the default shell
122 is \fB/usr/bin/sh\fR.
123 .RE

```

```

125 .sp
126 .LP
127 The maximum value of the \fIuid\fR and \fIgid\fR fields is \fB2147483647\fR. To
128 maximize interoperability and compatibility, administrators are recommended to
129 assign users a range of \fBUID\fRs and \fBGID\fRs below \fB60000\fR where
130 possible. (\fBUID\fRs from \fB0\fR-\fB99\fR inclusive are reserved by the
131 operating system vendor for use in future applications. Their use by end system
132 users or vendors of layered products is not supported and may cause security
133 related issues with future applications.)
134 .sp
135 .LP
136 The password file is an \fBASCII\fR file that resides in the \fB/etc\fR
137 directory. Because the encrypted passwords on a secure system are always kept
138 in the \fBshadow\fR file, \fB/etc/passwd\fR has general read permission on all
139 systems and can be used by routines that map between numerical user \fBID\fRs
140 and user names.
141 .sp
142 .LP
143 Blank lines are treated as malformed entries in the \fBpasswd\fR file and cause
144 consumers of the file, such as \fBgetpwnam\fR(3C), to fail.
145 .sp
146 .LP
147 The password file can contain entries beginning with a '+' (plus sign) or '-'
148 (minus sign) to selectively incorporate entries from another naming service
149 source, such as NIS, NIS+, or LDAP.
150 .sp
151 .LP
152 A line beginning with a '+' means to incorporate entries from the naming
153 service source. There are three styles of the '+' entries in this file. A
154 single + means to insert all the entries from the alternate naming service
155 source at that point, while a +\fIname\fR means to insert the specific entry,
156 if one exists, from the naming service source. A +@\fInetgroup\fR means to
157 insert the entries for all members of the network group \fInetgroup\fR from the
158 alternate naming service. If a +\fIname\fR entry has a non-null \fBpassword\fR,
159 \fBIgcos\fR, \fIhome-dir\fR, or \fIlogin-shell\fR field, the value of that field
160 overrides what is contained in the alternate naming service. The \fIuid\fR and
161 \fIgid\fR fields cannot be overridden.
162 .sp
163 .LP
164 A line beginning with a '\(mi' means to disallow entries from the alternate
165 naming service. There are two styles of '-' entries in this file. -\fIname\fR
166 means to disallow any subsequent entries (if any) for \fIname\fR (in this file
167 or in a naming service), and -@\fInetgroup\fR means to disallow any subsequent
168 entries for all members of the network group \fInetgroup\fR.
169 .sp
170 .LP
171 This is also supported by specifying 'compat' in
172 \fBnsswitch.conf\fR(4). The "compat" source might not be supported in future
173 releases. The preferred sources are \fBfiles\fR followed by the identifier of a
174 name service, such as \fBnis\fR or \fBldap\fR. This has the effect of
175 incorporating the entire contents of the naming service's \fBpasswd\fR database
176 or password-related information after the \fBpasswd\fR file.
177 .sp
178 .LP
179 Note that in compat mode, for every \fB/etc/passwd\fR entry, there must be a
180 corresponding entry in the \fB/etc/shadow\fR file.
181 .sp
182 .LP
183 Appropriate precautions must be taken to lock the \fB/etc/passwd\fR file
184 against simultaneous changes if it is to be edited with a text editor;
185 \fBvipw\fR(1B) does the necessary locking.
186 .SH EXAMPLES
187 .LP
188 \fBExample 1 \fRSample \fBpasswd\fR File
189 .sp
190 .LP

```

```

191 The following is a sample \fBpasswd\fR file:
192 .sp
193 .sp
194 .in +2
195 .nf
196 root:x:0:1:Super-User:/:/sbin/sh
197 fred:6k/7KCFRPNVXg:508:10:& Fredericks:/usr2/fred:/bin/csh
198 .fi
199 .in -2
200 .sp
201 .sp
202 .sp
203 .LP
204 and the sample password entry from \fBnsswitch.conf\fR:
205 .sp
206 .sp
207 .in +2
208 .nf
209 passwd: files ldap
210 .fi
211 .in -2
212 .sp
213 .sp
214 .sp
215 .LP
216 In this example, there are specific entries for users \fBroot\fR and \fBfred\fR
217 to assure that they can login even when the system is running single-user. In
218 addition, anyone whose password information is stored on an LDAP server will be
219 able to login with their usual password, shell, and home directory.
220 .sp
221 .sp
222 .LP
223 If the password file is:
224 .sp
225 .sp
226 .in +2
227 .nf
228 root:x:0:1:Super-User:/:/sbin/sh
229 fred:6k/7KCFRPNVXg:508:10:& Fredericks:/usr2/fred:/bin/csh
230 +
231 .fi
232 .in -2
233 .sp
234 .sp
235 .sp
236 .LP
237 and the password entry in \fBnsswitch.conf\fR is:
238 .sp
239 .sp
240 .in +2
241 .nf
242 passwd: compat
243 .fi
244 .in -2
245 .sp
246 .sp
247 .sp
248 .LP
249 then all the entries listed in the \fBNIS\fR \fBpasswd.byuid\fR and
250 \fBpasswd.byname\fR maps will be effectively incorporated after the entries for
251 \fBroot\fR and \fBfred\fR. If the password entry in \fBnsswitch.conf\fR is:
252 .sp
253 .sp
254 .in +2
255 .nf
256 passwd_compat: ldap

```

```

257 passwd: compat
258 .fi
259 .in -2

261 .sp
262 .LP
263 then all password-related entries stored on the LDAP server will be
264 incorporated after the entries for \fBroot\fR and \fBfred\fR.

266 .sp
267 .LP
268 The following is a sample \fBpasswd\fR file when \fBshadow\fR does not exist:

270 .sp
271 .in +2
272 .nf
273 root:q.mJzTnu8icf.:0:1:Super-User:/:/sbin/sh
274 fred:6k/7KCFRPNVXg:508:10:& Fredericks:/usr2/fred:/bin/csh
275 +john:
276 +@documentation:no-login:
277 +:::Guest
278 .fi
279 .in -2
280 .sp

282 .sp
283 .LP
284 The following is a sample \fBpasswd\fR file when \fBshadow\fR does exist:

286 .sp
287 .in +2
288 .nf
289 root:##root:0:1:Super-User:/:/sbin/sh
290 fred:##fred:508:10:& Fredericks:/usr2/fred:/bin/csh
291 +john:
292 +@documentation:no-login:
293 +:::Guest
294 .fi
295 .in -2
296 .sp

298 .sp
299 .LP
300 In this example, there are specific entries for users \fBroot\fR and
301 \fBfred\fR, to assure that they can log in even when the system is running
302 standalone. The user \fBjohn\fR will have his password entry in the naming
303 service source incorporated without change, anyone in the netgroup
304 \fBdocumentation\fR will have their password field disabled, and anyone else
305 will be able to log in with their usual password, shell, and home directory,
306 but with a \fBgcscos\fR field of \fBGuest\fR

308 .SH FILES
309 .sp
310 .ne 2
311 .na
312 \fB/etc/nsswitch.conf\fR
313 .ad
314 .RS 22n

316 .RE

318 .sp
319 .ne 2
320 .na
321 \fB/etc/passwd\fR
322 .ad

```

```

323 .RS 22n

325 .RE

327 .sp
328 .ne 2
329 .na
330 \fB/etc/shadow\fR
331 .ad
332 .RS 22n

334 .RE

336 .SH SEE ALSO
337 .sp
338 .LP
339 \fBchgrp\fR(1), \fBchown\fR(1), \fBfinger\fR(1), \fBgroups\fR(1),
340 \fBlogin\fR(1), \fBnewgrp\fR(1), \fBnispasswd\fR(1), \fBpasswd\fR(1),
341 \fBsh\fR(1), \fBsort\fR(1), \fBdomainname\fR(1M), \fBgetent\fR(1M),
342 \fBin.ftpd\fR(1M), \fBpassmgmt\fR(1M), \fBpwck\fR(1M), \fBpwconv\fR(1M),
343 \fBsu\fR(1M), \fBuseradd\fR(1M), \fBuserdel\fR(1M), \fBusermod\fR(1M),
344 \fBa64l\fR(3C), \fBcrypt\fR(3C), \fBgetpw\fR(3C), \fBgetpwnam\fR(3C),
345 \fBgetspnam\fR(3C), \fBputpwent\fR(3C), \fBgroup\fR(4), \fBhosts.equiv\fR(4),
346 \fBnsswitch.conf\fR(4), \fBshadow\fR(4), \fBenviron\fR(5),
347 \fBunistd.h\fR(3HEAD)
348 .sp
349 .LP
350 \fISystem Administration Guide: Basic Administration\fR

```

new/usr/src/pkg/manifests/SUNWcs.mf

1

```
*****
89641 Sat Jun 1 10:02:51 2013
new/usr/src/pkg/manifests/SUNWcs.mf
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 #
23 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
24 # Copyright 2012 Nexenta Systems, Inc. All rights reserved.
25 # Copyright (c) 2013 Gary Mills
26 #
27 #
28 <include SUNWcs.man1.inc>
29 <include SUNWcs.man1m.inc>
30 <include SUNWcs.man4.inc>
31 <include SUNWcs.man5.inc>
32 <include SUNWcs.man7d.inc>
33 <include SUNWcs.man7fs.inc>
34 set name=pkg.fmri value=pkg:/SUNWcs@$(PKGVERS)
35 set name=pkg.description \
36     value="core software for a specific instruction-set architecture"
37 set name=pkg.summary value="Core Solaris"
38 set name=info.classification value=org.opensolaris.category.2008:System/Core
39 set name=variant.arch value=$(ARCH)
40 dir path=dev group=sys
41 dir path=etc group=sys
42 dir path=etc/cron.d group=sys
43 dir path=etc/crypto group=sys
44 dir path=etc/crypto/certs group=sys
45 dir path=etc/crypto/crls group=sys
46 dir path=etc/default group=sys
47 dir path=etc/dev group=sys
48 dir path=etc/devices group=sys
49 dir path=etc/dfs group=sys
50 dir path=etc/dhcp group=sys
51 dir path=etc/fs group=sys
52 dir path=etc/fs/dev group=sys
53 dir path=etc/fs/hsfs group=sys
54 dir path=etc/fs/ufs group=sys
55 dir path=etc/ftpd group=sys
56 dir path=etc/inet group=sys
57 dir path=etc/init.d group=sys
58 dir path=etc/lib group=sys
59 dir path=etc/logadm.d group=sys
60 dir path=etc/mail group=mail
```

new/usr/src/pkg/manifests/SUNWcs.mf

2

```
61 dir path=etc/net group=sys
62 dir path=etc/net/ticlts group=sys
63 dir path=etc/net/ticots group=sys
64 dir path=etc/net/ticotsord group=sys
65 dir path=etc/opt group=sys
66 dir path=etc/rc0.d group=sys
67 dir path=etc/rc1.d group=sys
68 dir path=etc/rc2.d group=sys
69 dir path=etc/rc3.d group=sys
70 dir path=etc/rcS.d group=sys
71 dir path=etc/rpcsec group=sys
72 dir path=etc/saf
73 dir path=etc/saf/zsmon group=sys
74 dir path=etc/sasl group=sys
75 dir path=etc/security group=sys
76 dir path=etc/security/audit group=sys
77 dir path=etc/security/audit/localhost group=sys
78 dir path=etc/security/auth_attr.d group=sys
79 dir path=etc/security/dev group=sys
80 dir path=etc/security/exec_attr.d group=sys
81 dir path=etc/security/lib group=sys
82 dir path=etc/security/prof_attr.d group=sys
83 dir path=etc/skel group=sys
84 dir path=etc/svc group=sys
85 dir path=etc/svc/profile group=sys
86 dir path=etc/svc/profile/site group=sys
87 dir path=etc/svc/volatile group=sys
88 dir path=etc/sysevent group=sys
89 dir path=etc/sysevent/config group=sys
90 dir path=etc/tm group=sys
91 dir path=etc/user_attr.d group=sys
92 dir path=export group=sys
93 dir path=home group=root mode=0555
94 dir path=lib
95 dir path=lib/crypto
96 dir path=lib/inet
97 dir path=lib/svc
98 dir path=lib/svc/bin
99 dir path=lib/svc/capture
100 dir path=lib/svc/manifest group=sys
101 dir path=lib/svc/manifest/application group=sys
102 dir path=lib/svc/manifest/application/management group=sys
103 dir path=lib/svc/manifest/application/security group=sys
104 dir path=lib/svc/manifest/device group=sys
105 dir path=lib/svc/manifest/milestone group=sys
106 dir path=lib/svc/manifest/network group=sys
107 dir path=lib/svc/manifest/network/dns group=sys
108 dir path=lib/svc/manifest/network/ipsec group=sys
109 dir path=lib/svc/manifest/network/ldap group=sys
110 dir path=lib/svc/manifest/network/routing group=sys
111 dir path=lib/svc/manifest/network/rpc group=sys
112 dir path=lib/svc/manifest/network/shares group=sys
113 dir path=lib/svc/manifest/network/ssl group=sys
114 dir path=lib/svc/manifest/platform group=sys
115 $(sparc_ONLY)dir path=lib/svc/manifest/platform/sun4u group=sys
116 dir path=lib/svc/manifest/site group=sys
117 dir path=lib/svc/manifest/system group=sys
118 dir path=lib/svc/manifest/system/device group=sys
119 dir path=lib/svc/manifest/system/filesystem group=sys
120 dir path=lib/svc/manifest/system/security group=sys
121 dir path=lib/svc/manifest/system/svc group=sys
122 dir path=lib/svc/method
123 dir path=lib/svc/monitor
124 dir path=lib/svc/seed
125 dir path=lib/svc/share
126 dir path=mnt group=sys
```

```

127 dir path=opt group=sys
128 dir path=proc group=root mode=0555
129 dir path=root group=root mode=0700
130 dir path=sbin group=sys
131 dir path=system group=root
132 dir path=system/contract group=root mode=0555
133 dir path=system/object group=root mode=0555
134 dir path=tmp group=sys mode=1777
135 dir path=usr group=sys
136 dir path=usr/bin
137 dir path=usr/bin/$(ARCH32)
138 dir path=usr/bin/$(ARCH64)
139 dir path=usr/ccs
140 dir path=usr/ccs/bin
141 dir path=usr/demo
142 dir path=usr/games
143 dir path=usr/has
144 dir path=usr/has/bin
145 dir path=usr/has/lib
146 dir path=usr/has/man
147 dir path=usr/has/man/manlhas
148 dir path=usr/kernel group=sys
149 dir path=usr/kernel/drv group=sys
150 dir path=usr/kernel/drv/$(ARCH64) group=sys
151 dir path=usr/kernel/exec group=sys
152 dir path=usr/kernel/exec/$(ARCH64) group=sys
153 dir path=usr/kernel/fs group=sys
154 dir path=usr/kernel/fs/$(ARCH64) group=sys
155 dir path=usr/kernel/pcbe group=sys
156 dir path=usr/kernel/pcbe/$(ARCH64) group=sys
157 dir path=usr/kernel/sched group=sys
158 dir path=usr/kernel/sched/$(ARCH64) group=sys
159 dir path=usr/kernel/strmod group=sys
160 dir path=usr/kernel/strmod/$(ARCH64) group=sys
161 dir path=usr/kernel/sys group=sys
162 dir path=usr/kernel/sys/$(ARCH64) group=sys
163 dir path=usr/kvm
164 dir path=usr/lib
165 dir path=usr/lib/$(ARCH64)
166 dir path=usr/lib/audit
167 dir path=usr/lib/class
168 dir path=usr/lib/class/FX
169 dir path=usr/lib/class/IA
170 dir path=usr/lib/class/RT
171 dir path=usr/lib/class/SDC
172 dir path=usr/lib/class/TS
173 dir path=usr/lib/crypto
174 dir path=usr/lib/devfsadm group=sys
175 dir path=usr/lib/devfsadm/linkmod group=sys
176 dir path=usr/lib/fs group=sys
177 dir path=usr/lib/fs/autofs group=sys
178 dir path=usr/lib/fs/autofs/$(ARCH64) group=sys
179 dir path=usr/lib/fs/cachefs group=sys
180 dir path=usr/lib/fs/ctfs group=sys
181 dir path=usr/lib/fs/dev group=sys
182 dir path=usr/lib/fs/fd group=sys
183 dir path=usr/lib/fs/hsfs group=sys
184 dir path=usr/lib/fs/lofs group=sys
185 dir path=usr/lib/fs/mntfs group=sys
186 dir path=usr/lib/fs/nfs group=sys
187 dir path=usr/lib/fs/nfs/$(ARCH64) group=sys
188 dir path=usr/lib/fs/objfs group=sys
189 dir path=usr/lib/fs/proc group=sys
190 dir path=usr/lib/fs/sharefs group=sys
191 dir path=usr/lib/fs/tmpfs group=sys
192 dir path=usr/lib/fs/ufs group=sys

```

```

193 dir path=usr/lib/help
194 dir path=usr/lib/help/auths
195 dir path=usr/lib/help/auths/locale
196 dir path=usr/lib/help/auths/locale/C
197 dir path=usr/lib/help/profiles
198 dir path=usr/lib/help/profiles/locale
199 dir path=usr/lib/help/profiles/locale/C
200 dir path=usr/lib/iconv
201 dir path=usr/lib/inet
202 dir path=usr/lib/inet/$(ARCH32)
203 dir path=usr/lib/inet/$(ARCH64)
204 dir path=usr/lib/inet/dhcp
205 dir path=usr/lib/inet/dhcp/nsu
206 dir path=usr/lib/inet/dhcp/svc
207 dir path=usr/lib/locale
208 dir path=usr/lib/locale/C
209 dir path=usr/lib/locale/C/LC_COLLATE
210 dir path=usr/lib/locale/C/LC_CTYPE
211 dir path=usr/lib/locale/C/LC_MESSAGES
212 dir path=usr/lib/locale/C/LC_MONETARY
213 dir path=usr/lib/locale/C/LC_NUMERIC
214 dir path=usr/lib/locale/C/LC_TIME
215 dir path=usr/lib/netsvc group=sys
216 dir path=usr/lib/pci
217 dir path=usr/lib/rcm
218 dir path=usr/lib/rcm/modules
219 dir path=usr/lib/rcm/scripts
220 dir path=usr/lib/reparse
221 dir path=usr/lib/saf
222 dir path=usr/lib/secure
223 dir path=usr/lib/secure/$(ARCH64)
224 dir path=usr/lib/security
225 dir path=usr/lib/sysevent
226 dir path=usr/lib/sysevent/modules
227 dir path=usr/net group=sys
228 dir path=usr/net/nls group=sys
229 dir path=usr/net/servers group=sys
230 dir path=usr/old
231 dir path=usr/platform group=sys
232 dir path=usr/sadm
233 dir path=usr/sadm/bin
234 dir path=usr/sadm/install
235 dir path=usr/sadm/install/scripts
236 dir path=usr/sbin
237 $(i386_ONLY)dir path=usr/sbin/$(ARCH32)
238 dir path=usr/sbin/$(ARCH64)
239 dir path=usr/share
240 dir path=usr/share/doc group=other
241 dir path=usr/share/doc/ksh
242 dir path=usr/share/doc/ksh/images
243 dir path=usr/share/doc/ksh/images/callouts
244 dir path=usr/share/lib
245 dir path=usr/share/lib/mailx
246 dir path=usr/share/lib/pub
247 dir path=usr/share/lib/tabset
248 dir path=usr/share/lib/xml group=sys
249 dir path=usr/share/lib/xml/dtd group=sys
250 dir path=usr/share/lib/xml/style group=sys
251 dir path=usr/share/man
252 dir path=usr/share/man/man1
253 dir path=usr/share/man/man1m
254 dir path=usr/share/man/man4
255 dir path=usr/share/man/man5
256 dir path=usr/share/man/man7d
257 dir path=usr/share/man/man7fs
258 dir path=usr/share/src group=sys

```

```

259 dir path=var group=sys
260 dir path=var/adm/groupe=adm mode=0775
261 dir path=var/adm/exacct group=adm owner=adm
262 dir path=var/adm/log group=adm owner=adm
263 dir path=var/adm/streams group=sys
264 dir path=var/audit group=sys
265 dir path=var/cores group=sys
266 dir path=var/cron group=sys
267 dir path=var/games
268 dir path=var/ldmap group=daemon owner=daemon
269 dir path=var/inet group=sys
270 dir path=var/ld
271 dir path=var/ld/${ARCH64}
272 dir path=var/log group=sys
273 dir path=var/logadm
274 dir path=var/mail group=mail mode=1777
275 dir path=var/mail/:saved group=mail mode=0775
276 dir path=var/news
277 dir path=var/opt group=sys
278 dir path=var/preserve mode=1777
279 dir path=var/run group=sys
280 dir path=var/sadm group=sys
281 dir path=var/sadm/system group=sys
282 dir path=var/sadm/system/admin group=sys
283 dir path=var/saf
284 dir path=var/saf/zsmon group=sys
285 dir path=var/spool
286 dir path=var/spool/cron group=sys
287 dir path=var/spool/cron/atjobs group=sys
288 dir path=var/spool/cron/crontabs group=sys
289 dir path=var/spool/locks group=uucp owner=uucp
290 dir path=var/svc group=sys
291 dir path=var/svc/log group=sys
292 dir path=var/svc/manifest group=sys
293 dir path=var/svc/manifest/application group=sys
294 dir path=var/svc/manifest/application/management group=sys
295 dir path=var/svc/manifest/application/print group=sys
296 dir path=var/svc/manifest/application/security group=sys
297 dir path=var/svc/manifest/device group=sys
298 dir path=var/svc/manifest/milestone group=sys
299 dir path=var/svc/manifest/network group=sys
300 dir path=var/svc/manifest/network/dns group=sys
301 dir path=var/svc/manifest/network/ipsec group=sys
302 dir path=var/svc/manifest/network/ldap group=sys
303 dir path=var/svc/manifest/network/nfs group=sys
304 dir path=var/svc/manifest/network/nis group=sys
305 dir path=var/svc/manifest/network/routing group=sys
306 dir path=var/svc/manifest/network/rpc group=sys
307 dir path=var/svc/manifest/network/security group=sys
308 dir path=var/svc/manifest/network/shares group=sys
309 dir path=var/svc/manifest/network/ssl group=sys
310 dir path=var/svc/manifest/platform group=sys
311 $(sparc_ONLY)dir path=var/svc/manifest/platform/sun4u group=sys
312 $(sparc_ONLY)dir path=var/svc/manifest/platform/sun4v group=sys
313 dir path=var/svc/manifest/site group=sys
314 dir path=var/svc/manifest/system group=sys
315 dir path=var/svc/manifest/system/device group=sys
316 dir path=var/svc/manifest/system/filesystem group=sys
317 dir path=var/svc/manifest/system/security group=sys
318 dir path=var/svc/manifest/system/svc group=sys
319 dir path=var/svc/profile group=sys
320 dir path=var/tmp group=sys mode=1777
321 driver name=dump perms="dump 0660 root sys"
322 driver name=fsnap \
323     policy="ctl_read_priv_set=sys_config write_priv_set=sys_config" \
324     perms="* 0640 root sys" perms="ctl 0666 root sys"

```

```

325 driver name=kstat perms="* 0666 root sys"
326 driver name=ksyms perms="* 0666 root sys"
327 driver name=logindmux
328 driver name=ptm clone_perms="ptmx 0666 root sys"
329 driver name=pts perms="* 0644 root sys" perms="0 0620 root tty" \
330     perms="1 0620 root tty" perms="2 0620 root tty" perms="3 0620 root tty"
331 file path=etc/.login group=sys preserve=renamew
332 file path=etc/cron.d/.proto group=sys mode=0744
333 file path=etc/cron.d/at.deny group=sys preserve=true
334 file path=etc/cron.d/cron.deny group=sys preserve=true
335 file path=etc/cron.d/queuedefs group=sys
336 file path=etc/crypto/kmf.conf group=sys preserve=true
337 file path=etc/crypto/pkcs11.conf group=sys preserve=true
338 file path=etc/datensk group=sys mode=0444
339 file path=etc/default/cron group=sys preserve=true
340 file path=etc/default/devfsadm group=sys preserve=true
341 file path=etc/default/fs group=sys preserve=true
342 file path=etc/default/init group=sys preserve=true
343 file path=etc/default/keyserv group=sys preserve=true
344 file path=etc/default/login group=sys preserve=true
345 file path=etc/default/nss group=sys preserve=true
346 file path=etc/default/passwd group=sys preserve=true
347 file path=etc/default/su group=sys preserve=true
348 file path=etc/default/syslogd group=sys preserve=true
349 file path=etc/default/tar group=sys preserve=true
350 file path=etc/default/useradd group=sys preserve=true
351 file path=etc/default/utmpd group=sys preserve=true
352 file path=etc/dev/reserved_devnames group=sys preserve=true
353 file path=etc/device.tab group=root mode=0444 preserve=true
354 file path=etc/dfs/dfstab group=sys preserve=true
355 file path=etc/dfs/fstypes group=root preserve=true
356 file path=etc/dfs/sharetab group=root mode=0444 preserve=true
357 file path=etc/dgroup.tab group=sys mode=0444 preserve=true
358 file path=etc/dhcp/inittab group=sys preserve=true
359 file path=etc/dhcp/inittab6 group=sys preserve=true
360 file path=etc/dumpdates group=sys mode=0664 preserve=true
361 file path=etc/format.dat group=sys preserve=true
362 file path=etc/fs/dev/mount mode=0555
363 file path=etc/fs/hfs/mount mode=0555
364 file path=etc/fs/ufs/mount mode=0555
365 file path=etc/ftpd/ftpusers group=sys preserve=true
366 file path=etc/group group=sys preserve=true
367 file path=etc/inet/hosts group=sys preserve=true
368 file path=etc/inet/inetd.conf group=sys preserve=true
369 file path=etc/inet/ipaddrsel.conf group=sys preserve=true
370 file path=etc/inet/netmasks group=sys preserve=true
371 file path=etc/inet/networks group=sys preserve=true
372 file path=etc/inet/protocols group=sys preserve=true
373 file path=etc/inet/services group=sys preserve=true
374 file path=etc/inet/wanboot.conf.sample group=sys mode=0444
375 file path=etc/init.d/PRESERVE group=sys mode=0744 preserve=true
376 file path=etc/init.d/README group=sys preserve=true
377 file path=etc/init.d/cacheofs.daemon group=sys mode=0744 preserve=true
378 file path=etc/init.d/ldap.client group=sys mode=0744
379 file path=etc/init.d/nscd group=sys mode=0744
380 file path=etc/init.d/syssetup group=sys mode=0744 preserve=true
381 file path=etc/init.d/ufs_quota group=sys mode=0744 preserve=true
382 file path=etc/inittab group=sys preserve=true
383 file path=etc/ioctl.syscon group=sys preserve=true
384 file path=etc/ksh.kshrc group=sys preserve=renamew
385 file path=etc/logadm.conf group=sys preserve=true timestamp=19700101T000000Z
386 file path=etc/logindevperm group=sys preserve=true
387 file path=etc/magic mode=0444
388 file path=etc/mail/mailx.rc preserve=true
389 file path=etc/mailcap preserve=true
390 file path=etc/mime.types preserve=true

```

```

391 file path=etc/mnttab group=root mode=0444 preserve=true
392 file path=etc/motd group=sys preserve=true
393 file path=etc/net/ticlts/hosts group=sys
394 file path=etc/net/ticlts/services group=sys preserve=true
395 file path=etc/net/ticots/hosts group=sys
396 file path=etc/net/ticots/services group=sys preserve=true
397 file path=etc/net/ticotsord/hosts group=sys
398 file path=etc/net/ticotsord/services group=sys preserve=true
399 file path=etc/netconfig group=sys preserve=true
400 file path=etc/nsscd.conf group=sys preserve=true
401 file path=etc/nsswitch.ad group=sys
402 file path=etc/nsswitch.conf group=sys preserve=true
403 file path=etc/nsswitch.dns group=sys
404 file path=etc/nsswitch.files group=sys
405 file path=etc/nsswitch.ldap group=sys
406 file path=etc/pam.conf group=sys preserve=true
407 file path=etc/passwd group=sys preserve=true
408 file path=etc/profile group=sys preserve=true
409 file path=etc/project group=sys preserve=true
410 file path=etc/rc2.d/README group=sys
411 file path=etc/rc3.d/README group=sys
412 file path=etc/rc3.d/README group=sys
413 file path=etc/remote preserve=true
414 file path=etc/rpc group=sys preserve=true
415 file path=etc/saf/_sactab group=sys preserve=true
416 file path=etc/saf/_sysconfig group=sys preserve=true
417 file path=etc/saf/zsmon/_pmtab group=sys preserve=true
418 file path=etc/security/audit_class group=sys preserve=renamew
419 file path=etc/security/audit_event group=sys preserve=renamew
420 file path=etc/security/audit_warn group=sys mode=0740 preserve=renamew
421 file path=etc/security/auth_attr group=sys preserve=true \
422   timestamp=19700101T000000Z
423 file path=etc/security/auth_attr.d/SUNWcs group=sys
424 file path=etc/security/crypt.conf group=sys preserve=renamew
425 file path=etc/security/dev/audio mode=0400
426 file path=etc/security/dev/fd0 mode=0400
427 file path=etc/security/dev/sr0 mode=0400
428 file path=etc/security/dev/st0 mode=0400
429 file path=etc/security/dev/st1 mode=0400
430 file path=etc/security/exec_attr group=sys preserve=true \
431   timestamp=19700101T000000Z
432 file path=etc/security/exec_attr.d/SUNWcs group=sys
433 file path=etc/security/kmfpolicy.xml
434 file path=etc/security/lib/audio_clean group=sys mode=0555
435 file path=etc/security/lib/fd_clean group=sys mode=0555
436 file path=etc/security/lib/sr_clean group=sys mode=0555
437 file path=etc/security/lib/st_clean group=sys mode=0555
438 file path=etc/security/policy.conf group=sys preserve=true
439 file path=etc/security/priv_names group=sys preserve=renameold
440 file path=etc/security/prof_attr group=sys preserve=true \
441   timestamp=19700101T000000Z
442 file path=etc/security/prof_attr.d/SUNWcs group=sys
443 file path=etc/shadow group=sys mode=0400 preserve=true
444 file path=etc/skel/.profile group=other preserve=true
445 file path=etc/skel/local.cshrc group=sys preserve=true
446 file path=etc/skel/local.login group=sys preserve=true
447 file path=etc/skel/local.profile group=sys preserve=true
448 file path=etc/svc/profile/generic_limited_net.xml group=sys mode=0444
449 file path=etc/svc/profile/generic_open.xml group=sys mode=0444
450 file path=etc/svc/profile/inetd_generic.xml group=sys mode=0444
451 file path=etc/svc/profile/inetd_upgrade.xml group=sys mode=0444
452 file path=etc/svc/profile/ns_dns.xml group=sys mode=0444
453 file path=etc/svc/profile/ns_files.xml group=sys mode=0444
454 file path=etc/svc/profile/ns_ldap.xml group=sys mode=0444
455 file path=etc/svc/profile/ns_nis.xml group=sys mode=0444
456 file path=etc/svc/profile/ns_none.xml group=sys mode=0444

```

```

457 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,SPARC-Enterprise.xml \
458   group=sys mode=0444
459 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire-15000.xml \
460   group=sys mode=0444
461 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire-880.xml \
462   group=sys mode=0444
463 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire.xml group=sys \
464   mode=0444
465 $(sparc_ONLY)file \
466   path=etc/svc/profile/platform_SUNW,Ultra-Enterprise-10000.xml group=sys \
467   mode=0444
468 $(sparc_ONLY)file \
469   path=etc/svc/profile/platform_SUNW,UltraSPARC-III-Netrtract.xml group=sys \
470   mode=0444
471 file path=etc/svc/profile/platform_none.xml group=sys mode=0444
472 $(sparc_ONLY)file path=etc/svc/profile/platform_sun4v.xml group=sys mode=0444
473 file path=etc/sysevent/config/README group=sys mode=0444
474 file path=etc/sysevent/config/SUNW,EC_dr,ESC_dr_req,sysevent.conf group=sys
475 file path=etc/syslog.conf group=sys preserve=true
476 file path=etc/ttydefs group=sys preserve=true
477 file path=etc/ttysrch group=sys preserve=true
478 file path=etc/user_attr group=sys preserve=true timestamp=19700101T000000Z
479 file path=etc/user_attr.d/SUNWcs group=sys
480 file path=etc/vfstab group=sys preserve=true
481 file path=lib/inet/in.mpathd mode=0555
482 file path=lib/inet/ipmgmt mode=0555
483 file path=lib/inet/netcfgd mode=0555
484 file path=lib/inet/nwamd mode=0555
485 file path=lib/svc/bin/lsvcrun group=sys mode=0555
486 file path=lib/svc/bin/mfstscan group=sys mode=0555
487 file path=lib/svc/bin/restore_repository group=sys mode=0555
488 file path=lib/svc/bin/sqlite group=sys mode=0555
489 file path=lib/svc/bin/svc.configd group=sys mode=0555
490 file path=lib/svc/bin/svc.ipfd group=sys mode=0555
491 file path=lib/svc/bin/svc.startd group=sys mode=0555
492 file path=lib/svc/manifest/milestone/multi-user-server.xml group=sys mode=0444
493 file path=lib/svc/manifest/milestone/multi-user.xml group=sys mode=0444
494 file path=lib/svc/manifest/milestone/name-services.xml group=sys mode=0444
495 file path=lib/svc/manifest/milestone/network.xml group=sys mode=0444
496 file path=lib/svc/manifest/milestone/single-user.xml group=sys mode=0444
497 file path=lib/svc/manifest/milestone/sysconfig.xml group=sys mode=0444
498 file path=lib/svc/manifest/network/dlmgmt.xml group=sys mode=0444
499 file path=lib/svc/manifest/network/dns/client.xml group=sys mode=0444
500 file path=lib/svc/manifest/network/dns/install.xml group=sys mode=0444
501 file path=lib/svc/manifest/network/forwarding.xml group=sys mode=0444
502 file path=lib/svc/manifest/network/inetd-upgrade.xml group=sys mode=0444
503 file path=lib/svc/manifest/network/inetd.xml group=sys mode=0444
504 file path=lib/svc/manifest/network/ipsec/ike.xml group=sys mode=0444
505 file path=lib/svc/manifest/network/ipsec/ipsecalgs.xml group=sys mode=0444
506 file path=lib/svc/manifest/network/ipsec/manual-key.xml group=sys mode=0444
507 file path=lib/svc/manifest/network/ipsec/policy.xml group=sys mode=0444
508 file path=lib/svc/manifest/network/ldap/client.xml group=sys mode=0444
509 file path=lib/svc/manifest/network/network-initial.xml group=sys mode=0444
510 file path=lib/svc/manifest/network/network-install.xml group=sys mode=0444
511 file path=lib/svc/manifest/network/network-ipmgmt.xml group=sys mode=0444
512 file path=lib/svc/manifest/network/network-ipqos.xml group=sys mode=0444
513 file path=lib/svc/manifest/network/network-iptun.xml group=sys mode=0444
514 file path=lib/svc/manifest/network/network-location.xml group=sys mode=0444
515 file path=lib/svc/manifest/network/network-loopback.xml group=sys mode=0444
516 file path=lib/svc/manifest/network/network-netcfg.xml group=sys mode=0444
517 file path=lib/svc/manifest/network/network-netmask.xml group=sys mode=0444
518 file path=lib/svc/manifest/network/network-physical.xml group=sys mode=0444
519 file path=lib/svc/manifest/network/network-routing-setup.xml group=sys \
520   mode=0444
521 file path=lib/svc/manifest/network/network-service.xml group=sys mode=0444
522 file path=lib/svc/manifest/network/routing/legacy-routing.xml group=sys \

```



```

523 mode=0444
524 file path=lib/svc/manifest/network/rpc/bind.xml group=sys mode=0444
525 file path=lib/svc/manifest/network/rpc/keysevr.xml group=sys mode=0444
526 file path=lib/svc/manifest/network/shares/group.xml group=sys mode=0444
527 file path=lib/svc/manifest/network/shares/reparsed.xml group=sys mode=0444
528 file path=lib/svc/manifest/network/socket-filter-kssl.xml group=sys mode=0444
529 file path=lib/svc/manifest/network/ssl/kssl-proxy.xml group=sys mode=0444
530 file path=lib/svc/manifest/system/auditd.xml group=sys mode=0444
531 file path=lib/svc/manifest/system/auditset.xml group=sys mode=0444
532 file path=lib/svc/manifest/system/boot-archive-update.xml group=sys mode=0444
533 file path=lib/svc/manifest/system/boot-archive.xml group=sys mode=0444
534 file path=lib/svc/manifest/system/boot-config.xml group=sys mode=0444
535 file path=lib/svc/manifest/system/consadm.xml group=sys mode=0444
536 file path=lib/svc/manifest/system/console-login.xml group=sys mode=0444
537 file path=lib/svc/manifest/system/coreadm.xml group=sys mode=0444
538 file path=lib/svc/manifest/system/cron.xml group=sys mode=0444
539 file path=lib/svc/manifest/system/cryptosvc.xml group=sys mode=0444
540 file path=lib/svc/manifest/system/device/allocate.xml group=sys mode=0444
541 file path=lib/svc/manifest/system/device/devices-audio.xml group=sys mode=0444
542 file path=lib/svc/manifest/system/device/devices-local.xml group=sys mode=0444
543 file path=lib/svc/manifest/system/device/mpxio-upgrade.xml group=sys mode=0444
544 file path=lib/svc/manifest/system/early-manifest-import.xml group=sys \
545 mode=0444
546 file path=lib/svc/manifest/system/extended-accounting.xml group=sys mode=0444
547 file path=lib/svc/manifest/system/filesystem/local-fs.xml group=sys mode=0444
548 file path=lib/svc/manifest/system/filesystem/minimal-fs.xml group=sys \
549 mode=0444
550 file path=lib/svc/manifest/system/filesystem/root-fs.xml group=sys mode=0444
551 file path=lib/svc/manifest/system/filesystem/usr-fs.xml group=sys mode=0444
552 $(i386_ONLY)file path=lib/svc/manifest/system/hostid.xml group=sys mode=0444
553 file path=lib/svc/manifest/system/hotplug.xml group=sys mode=0444
554 file path=lib/svc/manifest/system/identity.xml group=sys mode=0444
555 file path=lib/svc/manifest/system/idmap.xml group=sys mode=0444
556 file path=lib/svc/manifest/system/keymap.xml group=sys mode=0444
557 file path=lib/svc/manifest/system/logadm-upgrade.xml group=sys mode=0444
558 file path=lib/svc/manifest/system/manifest-import.xml group=sys mode=0444
559 file path=lib/svc/manifest/system/name-service-cache.xml group=sys mode=0444
560 file path=lib/svc/manifest/system/pfexecd.xml group=sys mode=0444
561 file path=lib/svc/manifest/system/rbac.xml group=sys mode=0444
562 file path=lib/svc/manifest/system/rmtmpfiles.xml group=sys mode=0444
563 file path=lib/svc/manifest/system/sac.xml group=sys mode=0444
564 file path=lib/svc/manifest/system/svc/global.xml group=sys mode=0444
565 file path=lib/svc/manifest/system/svc/restarter.xml group=sys mode=0444
566 file path=lib/svc/manifest/system/system-log.xml group=sys mode=0444
567 file path=lib/svc/manifest/system/utmp.xml group=sys mode=0444
568 file path=lib/svc/manifest/system/vtdaemon.xml group=sys mode=0444
569 file path=lib/svc/method/boot-archive mode=0555
570 file path=lib/svc/method/boot-archive-update mode=0555
571 file path=lib/svc/method/console-login mode=0555
572 file path=lib/svc/method/devices-audio mode=0555
573 file path=lib/svc/method/devices-local mode=0555
574 file path=lib/svc/method/dns-install mode=0555
575 file path=lib/svc/method/fs-local mode=0555
576 file path=lib/svc/method/fs-minimal mode=0555
577 file path=lib/svc/method/fs-root mode=0555
578 file path=lib/svc/method/fs-usr mode=0555
579 file path=lib/svc/method/identity-domain mode=0555
580 file path=lib/svc/method/identity-node mode=0555
581 file path=lib/svc/method/inetd-upgrade mode=0555
582 file path=lib/svc/method/keymap mode=0555
583 file path=lib/svc/method/ldap-client mode=0555
584 file path=lib/svc/method/logadm-upgrade mode=0555
585 file path=lib/svc/method/manifest-import mode=0555
586 file path=lib/svc/method/mpxio-upgrade mode=0555
587 file path=lib/svc/method/net-init mode=0555
588 file path=lib/svc/method/net-install mode=0555

```

```

589 file path=lib/svc/method/net-ipmgmt mode=0555
590 file path=lib/svc/method/net-ipqos mode=0555
591 file path=lib/svc/method/net-iptun mode=0555
592 file path=lib/svc/method/net-loc mode=0555
593 file path=lib/svc/method/net-loopback mode=0555
594 file path=lib/svc/method/net-netmask mode=0555
595 file path=lib/svc/method/net-nwam mode=0555
596 file path=lib/svc/method/net-physical mode=0555
597 file path=lib/svc/method/net-routing-setup mode=0555
598 file path=lib/svc/method/net-svc mode=0555
599 file path=lib/svc/method/rmtmpfiles mode=0555
600 file path=lib/svc/method/rpc-bind mode=0555
601 file path=lib/svc/method/svc-allocate mode=0555
602 file path=lib/svc/method/svc-auditd mode=0555
603 file path=lib/svc/method/svc-auditset mode=0555
604 file path=lib/svc/method/svc-boot-config mode=0555
605 file path=lib/svc/method/svc-consadm mode=0555
606 file path=lib/svc/method/svc-cron mode=0555
607 file path=lib/svc/method/svc-dlmgmt mode=0555
608 file path=lib/svc/method/svc-forwarding mode=0555
609 $(i386_ONLY)file path=lib/svc/method/svc-hostid mode=0555
610 file path=lib/svc/method/svc-hotplug mode=0555
611 file path=lib/svc/method/svc-legacy-routing mode=0555
612 file path=lib/svc/method/svc-nscd mode=0555
613 file path=lib/svc/method/svc-rbac mode=0555
614 file path=lib/svc/method/svc-sockfilter mode=0555
615 file path=lib/svc/method/svc-utmpd mode=0555
616 file path=lib/svc/method/system-log mode=0555
617 file path=lib/svc/method/vtdaemon mode=0555
618 file path=lib/svc/method/yp mode=0555
619 # global.db is not needed in non-global zones, and it's pretty large.
620 file path=lib/svc/seed/global.db group=sys mode=0444 \
621 variant.opensolaris.zone=global
622 # symmetrically, nonglobal.db is not needed in global zones.
623 file path=lib/svc/seed/nonglobal.db group=sys mode=0444 \
624 variant.opensolaris.zone=nonglobal
625 file path=lib/svc/share/README mode=0444
626 file path=lib/svc/share/fs_include.sh mode=0444
627 file path=lib/svc/share/ipf_include.sh mode=0444
628 file path=lib/svc/share/mfsthistory mode=0444
629 file path=lib/svc/share/net_include.sh mode=0444
630 file path=lib/svc/share/routing_include.sh mode=0444
631 file path=lib/svc/share/smf_include.sh mode=0444
632 file path=root/.bashrc group=root preserve=true
633 file path=root/.profile group=root preserve=true
634 file path=sbin/autopush mode=0555
635 $(i386_ONLY)file path=sbin/biosdev mode=0555
636 file path=sbin/bootadm mode=0555
637 file path=sbin/cryptoadm mode=0555
638 file path=sbin/devprop mode=0555
639 file path=sbin/dhccpagent mode=0555
640 file path=sbin/dhccpinfo mode=0555
641 file path=sbin/dlmgmt mode=0555
642 file path=sbin/fdisk mode=0555
643 file path=sbin/fiocompress mode=0555
644 file path=sbin/hostconfig mode=0555
645 file path=sbin/ifconfig mode=0555
646 file path=sbin/ifparse mode=0555
647 file path=sbin/init group=sys mode=0555
648 $(i386_ONLY)file path=sbin/installgrub group=sys mode=0555
649 file path=sbin/impstat mode=0555
650 file path=sbin/mount mode=0555
651 file path=sbin/mountall group=sys mode=0555
652 file path=sbin/netstrategy mode=0555
653 file path=sbin/rc0 group=sys mode=0744
654 file path=sbin/rc1 group=sys mode=0744

```

```

655 file path=sbin/rc2 group=sys mode=0744
656 file path=sbin/rc3 group=sys mode=0744
657 file path=sbin/rcS group=sys mode=0744
658 file path=sbin/route mode=0555
659 file path=sbin/routeadm mode=0555
660 file path=sbin/soconfig mode=0555
661 file path=sbin/su.static group=sys mode=0555
662 file path=sbin/sulogin mode=0555
663 file path=sbin/swapadd group=sys mode=0744
664 file path=sbin/sync mode=0555
665 file path=sbin/tzreload mode=0555
666 file path=sbin/uadmin group=sys mode=0555
667 file path=sbin/umount mode=0555
668 file path=sbin/umountall group=sys mode=0555
669 file path=sbin/uname mode=0555
670 file path=sbin/wusbadm mode=0555
671 file path=sbin/zonename mode=0555
672 $(i386_ONLY)file path=usr/bin/$(ARCH32)/amt mode=0555
673 file path=usr/bin/$(ARCH32)/decrypt mode=0555
674 file path=usr/bin/$(ARCH32)/digest mode=0555
675 file path=usr/bin/$(ARCH32)/ksh93 mode=0555
676 $(i386_ONLY)file path=usr/bin/$(ARCH32)/newtask group=sys mode=4555
677 $(i386_ONLY)file path=usr/bin/$(ARCH32)/nohup mode=0555
678 $(i386_ONLY)file path=usr/bin/$(ARCH32)/prctl mode=0555
679 $(i386_ONLY)file path=usr/bin/$(ARCH32)/prstat mode=0555
680 $(i386_ONLY)file path=usr/bin/$(ARCH32)/ps mode=0555
681 file path=usr/bin/$(ARCH32)/savecore mode=0555
682 $(i386_ONLY)file path=usr/bin/$(ARCH32)/setuname mode=0555
683 $(i386_ONLY)file path=usr/bin/$(ARCH32)/uptime mode=4555
684 file path=usr/bin/$(ARCH64)/amt mode=0555
685 file path=usr/bin/$(ARCH64)/crle mode=0555
686 file path=usr/bin/$(ARCH64)/decrypt mode=0555
687 file path=usr/bin/$(ARCH64)/digest mode=0555
688 file path=usr/bin/$(ARCH64)/ksh93 mode=0555
689 file path=usr/bin/$(ARCH64)/ls mode=0555
690 file path=usr/bin/$(ARCH64)/moe mode=0555
691 file path=usr/bin/$(ARCH64)/newtask group=sys mode=4555
692 file path=usr/bin/$(ARCH64)/nohup mode=0555
693 file path=usr/bin/$(ARCH64)/prctl mode=0555
694 file path=usr/bin/$(ARCH64)/prstat mode=0555
695 file path=usr/bin/$(ARCH64)/ps mode=0555
696 file path=usr/bin/$(ARCH64)/savecore mode=0555
697 file path=usr/bin/$(ARCH64)/setuname mode=0555
698 file path=usr/bin/$(ARCH64)/uptime mode=4555
699 $(i386_ONLY)file path=usr/bin/addbadsec mode=0555
700 file path=usr/bin/alias mode=0555
701 file path=usr/bin/amt mode=0555
702 file path=usr/bin/arch mode=0555
703 file path=usr/bin/at group=sys mode=4755
704 file path=usr/bin/atq group=sys mode=4755
705 file path=usr/bin/atrm group=sys mode=4755
706 file path=usr/bin/auths mode=0555
707 file path=usr/bin/basename mode=0555
708 file path=usr/bin/busstat mode=0555
709 file path=usr/bin/captainfo mode=0555
710 file path=usr/bin/cat mode=0555
711 file path=usr/bin/chgrp mode=0555
712 file path=usr/bin/chmod mode=0555
713 file path=usr/bin/chown mode=0555
714 file path=usr/bin/ckdate mode=0555
715 file path=usr/bin/ckgid mode=0555
716 file path=usr/bin/ckint mode=0555
717 file path=usr/bin/ckitem mode=0555
718 file path=usr/bin/ckkeywd mode=0555
719 file path=usr/bin/ckpath mode=0555
720 file path=usr/bin/ckrange mode=0555

```

```

721 file path=usr/bin/ckstr mode=0555
722 file path=usr/bin/cktime mode=0555
723 file path=usr/bin/ckuid mode=0555
724 file path=usr/bin/ckyorn mode=0555
725 file path=usr/bin/clear mode=0555
726 file path=usr/bin/coreadm mode=0555
727 file path=usr/bin/cp mode=0555
728 file path=usr/bin/cpio mode=0555
729 file path=usr/bin/crle mode=0555
730 file path=usr/bin/crontab mode=4555
731 file path=usr/bin/crypt mode=0555
732 file path=usr/bin/csh mode=0555
733 file path=usr/bin/ctrunc mode=0555
734 file path=usr/bin/ctstat mode=0555
735 file path=usr/bin/ctwatch mode=0555
736 file path=usr/bin/date mode=0555
737 file path=usr/bin/dd mode=0555
738 file path=usr/bin/devattr mode=0555
739 file path=usr/bin/devfree mode=0555
740 file path=usr/bin/devreserv mode=0555
741 file path=usr/bin/dirname mode=0555
742 $(i386_ONLY)file path=usr/bin/diskscan mode=0555
743 file path=usr/bin/domainname mode=0555
744 file path=usr/bin/du mode=0555
745 file path=usr/bin/dumpcs mode=0555
746 file path=usr/bin/dumpkeys mode=0555
747 file path=usr/bin/echo mode=0555
748 file path=usr/bin/ed mode=0555
749 file path=usr/bin/egrep mode=0555
750 file path=usr/bin/eject mode=0555
751 file path=usr/bin/env mode=0555
752 file path=usr/bin/expr mode=0555
753 file path=usr/bin/false mode=0555
754 file path=usr/bin/fdetach mode=0555
755 file path=usr/bin/fdformat mode=4555
756 file path=usr/bin/fgrep mode=0555
757 file path=usr/bin/file mode=0555
758 file path=usr/bin/find mode=0555
759 file path=usr/bin/fmt mode=0555
760 file path=usr/bin/fmtmsg mode=0555
761 file path=usr/bin/fold mode=0555
762 file path=usr/bin/fsstat mode=0555
763 file path=usr/bin/geniconvtbl mode=0555
764 file path=usr/bin/getconf mode=0555
765 file path=usr/bin/getdev mode=0555
766 file path=usr/bin/getdgrp mode=0555
767 file path=usr/bin/getent mode=0555
768 file path=usr/bin/getfacl mode=0555
769 file path=usr/bin/getopt mode=0555
770 file path=usr/bin/gettext mode=0555
771 file path=usr/bin/getvol mode=0555
772 file path=usr/bin/grep mode=0555
773 file path=usr/bin/groups mode=0555
774 file path=usr/bin/head mode=0555
775 file path=usr/bin/hostid mode=0555
776 file path=usr/bin/hostname mode=0555
777 file path=usr/bin/i286 mode=0555
778 file path=usr/bin/iconv mode=0555
779 file path=usr/bin/id mode=0555
780 file path=usr/bin/infcmp mode=0555
781 file path=usr/bin/iostat mode=0555
782 file path=usr/bin/isainfo mode=0555
783 file path=usr/bin/isalist mode=0555
784 file path=usr/bin/kbd mode=0555
785 file path=usr/bin/keylogin mode=0555
786 file path=usr/bin/keylogout mode=0555

```

```

787 file path=usr/bin/kmfcfg mode=0555
788 file path=usr/bin/kvmstat mode=0555
789 file path=usr/bin/line mode=0555
790 file path=usr/bin/listdgrp mode=0555
791 file path=usr/bin/listusers mode=0555
792 file path=usr/bin/loadkeys mode=0555
793 file path=usr/bin/logger mode=0555
794 file path=usr/bin/login mode=4555
795 file path=usr/bin/logins mode=0750
796 file path=usr/bin/ls mode=0555
797 file path=usr/bin/m4 mode=0555
798 file path=usr/bin/mach mode=0555
799 file path=usr/bin/mail group=mail mode=2511
800 file path=usr/bin/mailx group=mail mode=2511
801 file path=usr/bin/makedev mode=0555
802 file path=usr/bin/mesg mode=0555
803 file path=usr/bin/mkdir mode=0555
804 file path=usr/bin/mkpwdict mode=0555
805 file path=usr/bin/mktemp mode=0555
806 file path=usr/bin/moe mode=0555
807 file path=usr/bin/more mode=0555
808 file path=usr/bin/mpstat mode=0555
809 file path=usr/bin/mt mode=0555
810 file path=usr/bin/netstat mode=0555
811 file path=usr/bin/newgrp group=sys mode=4755
812 file path=usr/bin/nice mode=0555
813 file path=usr/bin/optisa mode=0555
814 file path=usr/bin/pagesize mode=0555
815 file path=usr/bin/passwd group=sys mode=6555
816 file path=usr/bin/pathchk mode=0555
817 file path=usr/bin/pax mode=0555
818 file path=usr/bin/pfexec mode=0555
819 file path=usr/bin/pg mode=0555
820 file path=usr/bin/pgrep mode=0555
821 file path=usr/bin/pktool mode=0555
822 file path=usr/bin/pr mode=0555
823 file path=usr/bin/printf mode=0555
824 file path=usr/bin/priocntl mode=0555
825 file path=usr/bin/profiles mode=0555
826 file path=usr/bin/projects mode=0555
827 file path=usr/bin/putdev mode=0555
828 file path=usr/bin/putdgrp mode=0555
829 file path=usr/bin/pwd mode=0555
830 file path=usr/bin/renice mode=0555
831 file path=usr/bin/rm mode=0555
832 file path=usr/bin/rmdir mode=0555
833 file path=usr/bin/roles mode=0555
834 file path=usr/bin/rpcinfo mode=0555
835 file path=usr/bin/runat mode=0555
836 file path=usr/bin/script mode=0555
837 file path=usr/bin/sed mode=0555
838 file path=usr/bin/setfacl mode=0555
839 file path=usr/bin/setpgrp group=sys mode=0555
840 file path=usr/bin/settime mode=0555
841 file path=usr/bin/shcomp mode=0555
842 file path=usr/bin/strchg group=root mode=0555
843 file path=usr/bin/strconf group=root mode=0555
844 file path=usr/bin/stty mode=0555
845 file path=usr/bin/su group=sys mode=4555
846 file path=usr/bin/svcprop mode=0555
847 file path=usr/bin/svcs mode=0555
848 file path=usr/bin/tabs mode=0555
849 file path=usr/bin/tail mode=0555
850 file path=usr/bin/tic mode=0555
851 file path=usr/bin/time mode=0555
852 file path=usr/bin/tip mode=4511 owner=uucp

```

```

853 file path=usr/bin/tpmadm mode=0555
854 file path=usr/bin/tput mode=0555
855 file path=usr/bin/tr mode=0555
856 file path=usr/bin/true mode=0555
857 file path=usr/bin/tty mode=0555
858 file path=usr/bin/tzselect mode=0555
859 file path=usr/bin/userattr mode=0555
860 file path=usr/bin/vmstat mode=0555
861 file path=usr/bin/which mode=0555
862 file path=usr/bin/who mode=0555
863 file path=usr/bin/wracct mode=0555
864 file path=usr/bin/write group=tty mode=2555
865 file path=usr/bin/xargs mode=0555
866 file path=usr/bin/xstr mode=0555
867 file path=usr/has/bin/edit mode=0555
868 file path=usr/has/bin/sh mode=0555
869 file path=usr/has/man/manlhas/edit.lhas
870 file path=usr/has/man/manlhas/ex.lhas
871 file path=usr/has/man/manlhas/sh.lhas
872 file path=usr/has/man/manlhas/vi.lhas
873 file path=usr/kernel/drv/$(ARCH64)/dump group=sys
874 file path=usr/kernel/drv/$(ARCH64)/fssnap group=sys
875 file path=usr/kernel/drv/$(ARCH64)/kstat group=sys
876 file path=usr/kernel/drv/$(ARCH64)/ksyms group=sys
877 file path=usr/kernel/drv/$(ARCH64)/logindmux group=sys
878 file path=usr/kernel/drv/$(ARCH64)/ptm group=sys
879 file path=usr/kernel/drv/$(ARCH64)/pts group=sys
880 $(i386_ONLY)file path=usr/kernel/drv/dump group=sys
881 file path=usr/kernel/drv/dump.conf group=sys
882 $(i386_ONLY)file path=usr/kernel/drv/fssnap group=sys
883 file path=usr/kernel/drv/fssnap.conf group=sys
884 $(i386_ONLY)file path=usr/kernel/drv/kstat group=sys
885 file path=usr/kernel/drv/kstat.conf group=sys
886 $(i386_ONLY)file path=usr/kernel/drv/ksyms group=sys
887 file path=usr/kernel/drv/ksyms.conf group=sys
888 $(i386_ONLY)file path=usr/kernel/drv/logindmux group=sys
889 file path=usr/kernel/drv/logindmux.conf group=sys
890 $(i386_ONLY)file path=usr/kernel/drv/ptm group=sys
891 file path=usr/kernel/drv/ptm.conf group=sys
892 $(i386_ONLY)file path=usr/kernel/drv/pts group=sys
893 file path=usr/kernel/drv/pts.conf group=sys
894 file path=usr/kernel/exec/$(ARCH64)/javaexec group=sys mode=0755
895 file path=usr/kernel/exec/$(ARCH64)/shbinexec group=sys mode=0755
896 $(i386_ONLY)file path=usr/kernel/exec/javaexec group=sys mode=0755
897 $(i386_ONLY)file path=usr/kernel/exec/shbinexec group=sys mode=0755
898 file path=usr/kernel/fs/$(ARCH64)/fdfs group=sys mode=0755
899 file path=usr/kernel/fs/$(ARCH64)/pcfs group=sys mode=0755
900 $(i386_ONLY)file path=usr/kernel/fs/fdfs group=sys mode=0755
901 $(i386_ONLY)file path=usr/kernel/fs/pcfs group=sys mode=0755
902 file path=usr/kernel/sched/$(ARCH64)/FX group=sys mode=0755
903 file path=usr/kernel/sched/$(ARCH64)/FX_DPTBL group=sys mode=0755
904 file path=usr/kernel/sched/$(ARCH64)/IA group=sys mode=0755
905 file path=usr/kernel/sched/$(ARCH64)/RT group=sys mode=0755
906 file path=usr/kernel/sched/$(ARCH64)/RT_DPTBL group=sys mode=0755
907 $(i386_ONLY)file path=usr/kernel/sched/FX group=sys mode=0755
908 $(i386_ONLY)file path=usr/kernel/sched/FX_DPTBL group=sys mode=0755
909 $(i386_ONLY)file path=usr/kernel/sched/IA group=sys mode=0755
910 $(i386_ONLY)file path=usr/kernel/sched/RT group=sys mode=0755
911 $(i386_ONLY)file path=usr/kernel/sched/RT_DPTBL group=sys mode=0755
912 file path=usr/kernel/strmod/$(ARCH64)/cryptmod group=sys mode=0755
913 file path=usr/kernel/strmod/$(ARCH64)/rlmod group=sys mode=0755
914 file path=usr/kernel/strmod/$(ARCH64)/telmod group=sys mode=0755
915 $(i386_ONLY)file path=usr/kernel/strmod/cryptmod group=sys mode=0755
916 $(i386_ONLY)file path=usr/kernel/strmod/rlmod group=sys mode=0755
917 $(i386_ONLY)file path=usr/kernel/strmod/telmod group=sys mode=0755
918 file path=usr/kernel/sys/$(ARCH64)/acctctl group=sys mode=0755

```

```

919 file path=usr/kernel/sys/$(ARCH64)/exacctsys group=sys mode=0755
920 file path=usr/kernel/sys/$(ARCH64)/sysacct group=sys mode=0755
921 $(i386_ONLY)file path=usr/kernel/sys/acctctl group=sys mode=0755
922 $(i386_ONLY)file path=usr/kernel/sys/exacctsys group=sys mode=0755
923 $(i386_ONLY)file path=usr/kernel/sys/sysacct group=sys mode=0755
924 file path=usr/kvm/README group=sys
925 file path=usr/lib/$(ARCH64)/libshare.so.1
926 file path=usr/lib/audit/audit_record_attr mode=0444
927 file path=usr/lib/calprog mode=0555
928 file path=usr/lib/class/FX/FXdispadmin mode=0555
929 file path=usr/lib/class/FX/FXprioctl mode=0555
930 file path=usr/lib/class/IA/IAdispadmin mode=0555
931 file path=usr/lib/class/IA/IAprioctl mode=0555
932 file path=usr/lib/class/RT/RTdispadmin mode=0555
933 file path=usr/lib/class/RT/RTprioctl mode=0555
934 file path=usr/lib/class/SDC/SDCdispadmin mode=0555
935 file path=usr/lib/class/SDC/SDCprioctl mode=0555
936 file path=usr/lib/class/TS/TSdispadmin mode=0555
937 file path=usr/lib/class/TS/TSprioctl mode=0555
938 file path=usr/lib/devfsadm/linkmod/SUNW_audio_link.so group=sys
939 file path=usr/lib/devfsadm/linkmod/SUNW_cfg_link.so group=sys
940 file path=usr/lib/devfsadm/linkmod/SUNW_disk_link.so group=sys
941 file path=usr/lib/devfsadm/linkmod/SUNW_fssnap_link.so group=sys
942 file path=usr/lib/devfsadm/linkmod/SUNW_ieee1394_link.so group=sys
943 file path=usr/lib/devfsadm/linkmod/SUNW_lofi_link.so group=sys
944 file path=usr/lib/devfsadm/linkmod/SUNW_md_link.so group=sys
945 file path=usr/lib/devfsadm/linkmod/SUNW_misc_link.so group=sys
946 file path=usr/lib/devfsadm/linkmod/SUNW_misc_link_$(ARCH).so group=sys
947 file path=usr/lib/devfsadm/linkmod/SUNW_port_link.so group=sys
948 file path=usr/lib/devfsadm/linkmod/SUNW_ramdisk_link.so group=sys
949 file path=usr/lib/devfsadm/linkmod/SUNW_sgen_link.so group=sys
950 file path=usr/lib/devfsadm/linkmod/SUNW_smp_link.so group=sys
951 file path=usr/lib/devfsadm/linkmod/SUNW_tape_link.so group=sys
952 file path=usr/lib/devfsadm/linkmod/SUNW_usb_link.so group=sys
953 $(i386_ONLY)file path=usr/lib/devfsadm/linkmod/SUNW_xen_link.so group=sys
954 file path=usr/lib/diffh mode=0555
955 file path=usr/lib/expreserve mode=0555
956 file path=usr/lib/exrecover mode=0555
957 file path=usr/lib/fs/cachefs/cachefsd mode=0555
958 file path=usr/lib/fs/cachefs/cachefslog mode=0555
959 file path=usr/lib/fs/cachefs/cachefspack mode=0555
960 file path=usr/lib/fs/cachefs/cachefsstat mode=0555
961 file path=usr/lib/fs/cachefs/cachefswssize mode=0555
962 file path=usr/lib/fs/cachefs/cfsadmin mode=0555
963 file path=usr/lib/fs/cachefs/cfsfstype mode=0555
964 file path=usr/lib/fs/cachefs/cfstagchk mode=0555
965 file path=usr/lib/fs/cachefs/dfshares mode=0555
966 file path=usr/lib/fs/cachefs/fsck mode=0555
967 file path=usr/lib/fs/cachefs/mount mode=0555
968 file path=usr/lib/fs/cachefs/share mode=0555
969 file path=usr/lib/fs/cachefs/umount mode=0555
970 file path=usr/lib/fs/cachefs/unshare mode=0555
971 file path=usr/lib/fs/ctfs/mount mode=0555
972 file path=usr/lib/fs/fd/mount mode=0555
973 file path=usr/lib/fs/hsfs/fstyp.so.1 mode=0555
974 file path=usr/lib/fs/hsfs/labelit mode=0555
975 file path=usr/lib/fs/lofs/mount mode=0555
976 file path=usr/lib/fs/mntfs/mount mode=0555
977 file path=usr/lib/fs/objfs/mount mode=0555
978 file path=usr/lib/fs/proc/mount mode=0555
979 file path=usr/lib/fs/sharefs/mount mode=0555
980 file path=usr/lib/fs/tmpfs/mount mode=0555
981 file path=usr/lib/fs/ufs/clri mode=0555
982 file path=usr/lib/fs/ufs/df mode=0555
983 file path=usr/lib/fs/ufs/edquota mode=0555
984 file path=usr/lib/fs/ufs/ff mode=0555

```

```

985 file path=usr/lib/fs/ufs/fsck mode=0555
986 file path=usr/lib/fs/ufs/fsckall mode=0555
987 file path=usr/lib/fs/ufs/fsdb mode=0555
988 file path=usr/lib/fs/ufs/fsirand mode=0555
989 file path=usr/lib/fs/ufs/fssnap mode=0555
990 file path=usr/lib/fs/ufs/fstyp.so.1 mode=0555
991 file path=usr/lib/fs/ufs/labelit mode=0555
992 file path=usr/lib/fs/ufs/lockfs mode=0555
993 file path=usr/lib/fs/ufs/mkfs mode=0555
994 file path=usr/lib/fs/ufs/ncheck mode=0555
995 file path=usr/lib/fs/ufs/newfs mode=0555
996 file path=usr/lib/fs/ufs/quot mode=0555
997 file path=usr/lib/fs/ufs/quota mode=4555
998 file path=usr/lib/fs/ufs/quotacheck mode=0555
999 file path=usr/lib/fs/ufs/quotaooff mode=0555
1000 file path=usr/lib/fs/ufs/repquota mode=0555
1001 file path=usr/lib/fs/ufs/tunefs mode=0555
1002 file path=usr/lib/fs/ufs/ufsdump mode=4555
1003 file path=usr/lib/fs/ufs/ufsrestore mode=4555
1004 file path=usr/lib/fs/ufs/volcopy mode=0555
1005 file path=usr/lib/getoptcvt mode=0555
1006 file path=usr/lib/help/auths/locale/C/AllSolAuthsHeader.html
1007 file path=usr/lib/help/auths/locale/C/AuditHeader.html
1008 file path=usr/lib/help/auths/locale/C/AuthJobsAdmin.html
1009 file path=usr/lib/help/auths/locale/C/AuthJobsUser.html
1010 file path=usr/lib/help/auths/locale/C/AuthProfmgrAssign.html
1011 file path=usr/lib/help/auths/locale/C/AuthProfmgrDelegate.html
1012 file path=usr/lib/help/auths/locale/C/AuthProfmgrExecattrWrite.html
1013 file path=usr/lib/help/auths/locale/C/AuthProfmgrRead.html
1014 file path=usr/lib/help/auths/locale/C/AuthProfmgrWrite.html
1015 file path=usr/lib/help/auths/locale/C/AuthReadNDMP.html
1016 file path=usr/lib/help/auths/locale/C/AuthReadSMB.html
1017 file path=usr/lib/help/auths/locale/C/AuthRoleAssign.html
1018 file path=usr/lib/help/auths/locale/C/AuthRoleDelegate.html
1019 file path=usr/lib/help/auths/locale/C/AuthRoleWrite.html
1020 file path=usr/lib/help/auths/locale/C/BindStates.html
1021 file path=usr/lib/help/auths/locale/C/DevAllocHeader.html
1022 file path=usr/lib/help/auths/locale/C/DevAllocate.html
1023 file path=usr/lib/help/auths/locale/C/DevConfig.html
1024 file path=usr/lib/help/auths/locale/C/DevGrant.html
1025 file path=usr/lib/help/auths/locale/C/DevRevoke.html
1026 file path=usr/lib/help/auths/locale/C/DhccpMgrHeader.html
1027 file path=usr/lib/help/auths/locale/C/DhccpMgrWrite.html
1028 file path=usr/lib/help/auths/locale/C/HotplugHeader.html
1029 file path=usr/lib/help/auths/locale/C/HotplugModify.html
1030 file path=usr/lib/help/auths/locale/C/IdmapRules.html
1031 file path=usr/lib/help/auths/locale/C/JobHeader.html
1032 file path=usr/lib/help/auths/locale/C/JobsGrant.html
1033 file path=usr/lib/help/auths/locale/C/LinkSecurity.html
1034 file path=usr/lib/help/auths/locale/C/LoginEnable.html
1035 file path=usr/lib/help/auths/locale/C/LoginHeader.html
1036 file path=usr/lib/help/auths/locale/C/LoginRemote.html
1037 file path=usr/lib/help/auths/locale/C/NetworkAutoconfRead.html
1038 file path=usr/lib/help/auths/locale/C/NetworkAutoconfSelect.html
1039 file path=usr/lib/help/auths/locale/C/NetworkAutoconfWlan.html
1040 file path=usr/lib/help/auths/locale/C/NetworkAutoconfWrite.html
1041 file path=usr/lib/help/auths/locale/C/NetworkHeader.html
1042 file path=usr/lib/help/auths/locale/C/NetworkILBconf.html
1043 file path=usr/lib/help/auths/locale/C/NetworkILBenable.html
1044 file path=usr/lib/help/auths/locale/C/NetworkAutoconfWlan.config.html
1045 file path=usr/lib/help/auths/locale/C/NetworkVRRP.html
1046 file path=usr/lib/help/auths/locale/C/PriAdmin.html
1047 file path=usr/lib/help/auths/locale/C/ProfmgrHeader.html
1048 file path=usr/lib/help/auths/locale/C/RoleHeader.html
1049 file path=usr/lib/help/auths/locale/C/SmfAllocate.html
1050 file path=usr/lib/help/auths/locale/C/SmfAutofsStates.html

```

1051 file path=usr/lib/help/auths/locale/C/SmfCoreadmStates.html  
 1052 file path=usr/lib/help/auths/locale/C/SmfCronStates.html  
 1053 file path=usr/lib/help/auths/locale/C/SmfExAcctFlowStates.html  
 1054 file path=usr/lib/help/auths/locale/C/SmfExAcctNetStates.html  
 1055 file path=usr/lib/help/auths/locale/C/SmfExAcctProcessStates.html  
 1056 file path=usr/lib/help/auths/locale/C/SmfExAcctTaskStates.html  
 1057 file path=usr/lib/help/auths/locale/C/SmfHeader.html  
 1058 file path=usr/lib/help/auths/locale/C/SmfILBStates.html  
 1059 file path=usr/lib/help/auths/locale/C/SmfIPsecStates.html  
 1060 file path=usr/lib/help/auths/locale/C/SmfIdmapStates.html  
 1061 file path=usr/lib/help/auths/locale/C/SmfInetdStates.html  
 1062 file path=usr/lib/help/auths/locale/C/SmfLocationStates.html  
 1063 file path=usr/lib/help/auths/locale/C/SmfMDNSStates.html  
 1064 file path=usr/lib/help/auths/locale/C/SmfManageAudit.html  
 1065 file path=usr/lib/help/auths/locale/C/SmfManageHeader.html  
 1066 file path=usr/lib/help/auths/locale/C/SmfManageHotplug.html  
 1067 file path=usr/lib/help/auths/locale/C/SmfManageZFSSnap.html  
 1068 file path=usr/lib/help/auths/locale/C/SmfModifyAppl.html  
 1069 file path=usr/lib/help/auths/locale/C/SmfModifyDepend.html  
 1070 file path=usr/lib/help/auths/locale/C/SmfModifyFramework.html  
 1071 file path=usr/lib/help/auths/locale/C/SmfModifyHeader.html  
 1072 file path=usr/lib/help/auths/locale/C/SmfModifyMethod.html  
 1073 file path=usr/lib/help/auths/locale/C/SmfNADDStates.html  
 1074 file path=usr/lib/help/auths/locale/C/SmfNDMPStates.html  
 1075 file path=usr/lib/help/auths/locale/C/SmfNWAMStates.html  
 1076 file path=usr/lib/help/auths/locale/C/SmfNscdStates.html  
 1077 file path=usr/lib/help/auths/locale/C/SmfPowerStates.html  
 1078 file path=usr/lib/help/auths/locale/C/SmfReparseStates.html  
 1079 file path=usr/lib/help/auths/locale/C/SmfRoutingStates.html  
 1080 file path=usr/lib/help/auths/locale/C/SmfSMBFSStates.html  
 1081 file path=usr/lib/help/auths/locale/C/SmfSMBStates.html  
 1082 file path=usr/lib/help/auths/locale/C/SmfSendmailStates.html  
 1083 file path=usr/lib/help/auths/locale/C/SmfSshStates.html  
 1084 file path=usr/lib/help/auths/locale/C/SmfSyslogStates.html  
 1085 file path=usr/lib/help/auths/locale/C/SmfVRRPStates.html  
 1086 file path=usr/lib/help/auths/locale/C/SmfValueAudit.html  
 1087 file path=usr/lib/help/auths/locale/C/SmfValueCoreadm.html  
 1088 file path=usr/lib/help/auths/locale/C/SmfValueExAcctFlow.html  
 1089 file path=usr/lib/help/auths/locale/C/SmfValueExAcctNet.html  
 1090 file path=usr/lib/help/auths/locale/C/SmfValueExAcctProcess.html  
 1091 file path=usr/lib/help/auths/locale/C/SmfValueExAcctTask.html  
 1092 file path=usr/lib/help/auths/locale/C/SmfValueFirewall.html  
 1093 file path=usr/lib/help/auths/locale/C/SmfValueHeader.html  
 1094 file path=usr/lib/help/auths/locale/C/SmfValueIPsec.html  
 1095 file path=usr/lib/help/auths/locale/C/SmfValueIdmap.html  
 1096 file path=usr/lib/help/auths/locale/C/SmfValueInetd.html  
 1097 file path=usr/lib/help/auths/locale/C/SmfValueMDNS.html  
 1098 file path=usr/lib/help/auths/locale/C/SmfValueNADD.html  
 1099 file path=usr/lib/help/auths/locale/C/SmfValueNDMP.html  
 1100 file path=usr/lib/help/auths/locale/C/SmfValueNWAM.html  
 1101 file path=usr/lib/help/auths/locale/C/SmfValueRouting.html  
 1102 file path=usr/lib/help/auths/locale/C/SmfValueSMB.html  
 1103 file path=usr/lib/help/auths/locale/C/SmfValueVscan.html  
 1104 file path=usr/lib/help/auths/locale/C/SmfValueVt.html  
 1105 file path=usr/lib/help/auths/locale/C/SmfVscanStates.html  
 1106 file path=usr/lib/help/auths/locale/C/SmfVtStates.html  
 1107 file path=usr/lib/help/auths/locale/C/SmfWpaStates.html  
 1108 file path=usr/lib/help/auths/locale/C/SysCpuPowerMgmt.html  
 1109 file path=usr/lib/help/auths/locale/C/SysDate.html  
 1110 file path=usr/lib/help/auths/locale/C/SysHeader.html  
 1111 file path=usr/lib/help/auths/locale/C/SysMaintenance.html  
 1112 file path=usr/lib/help/auths/locale/C/SysPowerMgmtBrightness.html  
 1113 file path=usr/lib/help/auths/locale/C/SysPowerMgmtHeader.html  
 1114 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspend.html  
 1115 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspendtoDisk.html  
 1116 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspendtoRAM.html

1117 file path=usr/lib/help/auths/locale/C/SysShutdown.html  
 1118 file path=usr/lib/help/auths/locale/C/SysSyseventRead.html  
 1119 file path=usr/lib/help/auths/locale/C/SysSyseventWrite.html  
 1120 file path=usr/lib/help/auths/locale/C/WifiConfig.html  
 1121 file path=usr/lib/help/auths/locale/C/WifiWep.html  
 1122 file path=usr/lib/help/auths/locale/C/ZoneCloneFrom.html  
 1123 file path=usr/lib/help/auths/locale/C/ZoneHeader.html  
 1124 file path=usr/lib/help/auths/locale/C/ZoneLogin.html  
 1125 file path=usr/lib/help/auths/locale/C/ZoneManage.html  
 1126 file path=usr/lib/help/profiles/locale/C/RtAcctadm.html  
 1127 file path=usr/lib/help/profiles/locale/C/RtAll.html  
 1128 file path=usr/lib/help/profiles/locale/C/RtAuditCfg.html  
 1129 file path=usr/lib/help/profiles/locale/C/RtAuditCtrl.html  
 1130 file path=usr/lib/help/profiles/locale/C/RtAuditReview.html  
 1131 file path=usr/lib/help/profiles/locale/C/RtCPUPowerManagement.html  
 1132 file path=usr/lib/help/profiles/locale/C/RtConsUser.html  
 1133 file path=usr/lib/help/profiles/locale/C/RtContractObserver.html  
 1134 file path=usr/lib/help/profiles/locale/C/RtCronMngmnt.html  
 1135 file path=usr/lib/help/profiles/locale/C/RtCryptoMngmnt.html  
 1136 file path=usr/lib/help/profiles/locale/C/RtDHCPMngmnt.html  
 1137 file path=usr/lib/help/profiles/locale/C/RtDatAdmin.html  
 1138 file path=usr/lib/help/profiles/locale/C/RtDefault.html  
 1139 file path=usr/lib/help/profiles/locale/C/RtDeviceMngmnt.html  
 1140 file path=usr/lib/help/profiles/locale/C/RtDeviceSecurity.html  
 1141 file path=usr/lib/help/profiles/locale/C/RtExAcctFlow.html  
 1142 file path=usr/lib/help/profiles/locale/C/RtExAcctNet.html  
 1143 file path=usr/lib/help/profiles/locale/C/RtExAcctProcess.html  
 1144 file path=usr/lib/help/profiles/locale/C/RtExAcctTask.html  
 1145 file path=usr/lib/help/profiles/locale/C/RtFTPmngmnt.html  
 1146 file path=usr/lib/help/profiles/locale/C/RtFileSysMngmnt.html  
 1147 file path=usr/lib/help/profiles/locale/C/RtFileSysSecurity.html  
 1148 file path=usr/lib/help/profiles/locale/C/RtHotplugMngmnt.html  
 1149 file path=usr/lib/help/profiles/locale/C/RtIPFilterMngmnt.html  
 1150 file path=usr/lib/help/profiles/locale/C/RtIdmapMngmnt.html  
 1151 file path=usr/lib/help/profiles/locale/C/RtIdmapNameRulesMngmnt.html  
 1152 file path=usr/lib/help/profiles/locale/C/RtInetdMngmnt.html  
 1153 file path=usr/lib/help/profiles/locale/C/RtKerberosClnMngmnt.html  
 1154 file path=usr/lib/help/profiles/locale/C/RtKerberosSrvrMngmnt.html  
 1155 file path=usr/lib/help/profiles/locale/C/RtLogMngmnt.html  
 1156 file path=usr/lib/help/profiles/locale/C/RtMailMngmnt.html  
 1157 file path=usr/lib/help/profiles/locale/C/RtMaintAndRepair.html  
 1158 file path=usr/lib/help/profiles/locale/C/RtMediaBkup.html  
 1159 file path=usr/lib/help/profiles/locale/C/RtMediaCtlg.html  
 1160 file path=usr/lib/help/profiles/locale/C/RtMediaRestore.html  
 1161 file path=usr/lib/help/profiles/locale/C/RtNDMPMngmnt.html  
 1162 file path=usr/lib/help/profiles/locale/C/RtNameServiceAdmin.html  
 1163 file path=usr/lib/help/profiles/locale/C/RtNameServiceSecure.html  
 1164 file path=usr/lib/help/profiles/locale/C/RtNetAutoconfAdmin.html  
 1165 file path=usr/lib/help/profiles/locale/C/RtNetAutoconfUser.html  
 1166 file path=usr/lib/help/profiles/locale/C/RtNetILB.html  
 1167 file path=usr/lib/help/profiles/locale/C/RtNetIPsec.html  
 1168 file path=usr/lib/help/profiles/locale/C/RtNetLinkSecure.html  
 1169 file path=usr/lib/help/profiles/locale/C/RtNetMngmnt.html  
 1170 file path=usr/lib/help/profiles/locale/C/RtNetObservability.html  
 1171 file path=usr/lib/help/profiles/locale/C/RtNetSecure.html  
 1172 file path=usr/lib/help/profiles/locale/C/RtNetVRRP.html  
 1173 file path=usr/lib/help/profiles/locale/C/RtNetWifiMngmnt.html  
 1174 file path=usr/lib/help/profiles/locale/C/RtNetWifiSecure.html  
 1175 file path=usr/lib/help/profiles/locale/C/RtObAccessMngmnt.html  
 1176 file path=usr/lib/help/profiles/locale/C/RtOperator.html  
 1177 file path=usr/lib/help/profiles/locale/C/RtPriAdmin.html  
 1178 file path=usr/lib/help/profiles/locale/C/RtPrntAdmin.html  
 1179 file path=usr/lib/help/profiles/locale/C/RtProcManagement.html  
 1180 file path=usr/lib/help/profiles/locale/C/RtReservedProfile.html  
 1181 file path=usr/lib/help/profiles/locale/C/RtRightsDelegate.html  
 1182 file path=usr/lib/help/profiles/locale/C/RtRightsDelegate.html

```

1183 file path=usr/lib/help/profiles/locale/C/RtSMBFSMngmnt.html
1184 file path=usr/lib/help/profiles/locale/C/RtSMBMngmnt.html
1185 file path=usr/lib/help/profiles/locale/C/RtSoftwareInstall.html
1186 file path=usr/lib/help/profiles/locale/C/RtSysAdmin.html
1187 file path=usr/lib/help/profiles/locale/C/RtSysEvMngmnt.html
1188 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmt.html
1189 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtBrightness.html
1190 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspend.html
1191 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspendtoDisk.html
1192 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspendtoRAM.html
1193 file path=usr/lib/help/profiles/locale/C/RtUserMngmnt.html
1194 file path=usr/lib/help/profiles/locale/C/RtUserSecurity.html
1195 file path=usr/lib/help/profiles/locale/C/RtVscanMngmnt.html
1196 file path=usr/lib/help/profiles/locale/C/RtZFSFileSysMngmnt.html
1197 file path=usr/lib/help/profiles/locale/C/RtZFSStorageMngmnt.html
1198 file path=usr/lib/help/profiles/locale/C/RtZoneMngmnt.html
1199 file path=usr/lib/help/profiles/locale/C/RtZoneSecurity.html
1200 file path=usr/lib/hotplugd mode=0555
1201 file path=usr/lib/iconv/646da.8859.t mode=0444
1202 file path=usr/lib/iconv/646de.8859.t mode=0444
1203 file path=usr/lib/iconv/646en.8859.t mode=0444
1204 file path=usr/lib/iconv/646es.8859.t mode=0444
1205 file path=usr/lib/iconv/646fr.8859.t mode=0444
1206 file path=usr/lib/iconv/646it.8859.t mode=0444
1207 file path=usr/lib/iconv/646sv.8859.t mode=0444
1208 file path=usr/lib/iconv/8859.646.t mode=0444
1209 file path=usr/lib/iconv/8859.646da.t mode=0444
1210 file path=usr/lib/iconv/8859.646de.t mode=0444
1211 file path=usr/lib/iconv/8859.646en.t mode=0444
1212 file path=usr/lib/iconv/8859.646es.t mode=0444
1213 file path=usr/lib/iconv/8859.646fr.t mode=0444
1214 file path=usr/lib/iconv/8859.646it.t mode=0444
1215 file path=usr/lib/iconv/8859.646sv.t mode=0444
1216 file path=usr/lib/iconv/iconv_data mode=0444
1217 file path=usr/lib/idmapd mode=0555
1218 file path=usr/lib/inet/${ARCH32}/in.iked mode=0555
1219 file path=usr/lib/inet/${ARCH64}/in.iked mode=0555
1220 file path=usr/lib/inet/certdb mode=0555
1221 file path=usr/lib/inet/certlocal mode=0555
1222 file path=usr/lib/inet/certrldb mode=0555
1223 file path=usr/lib/inet/inetd mode=0555
1224 file path=usr/lib/intrd mode=0555
1225 file path=usr/lib/isaexec mode=0555
1226 file path=usr/lib/ksladm mode=0555
1227 $(sparc_ONLY)file path=usr/lib/ld.so
1228 file path=usr/lib/libshare.so.1
1229 file path=usr/lib/makekey mode=0555
1230 file path=usr/lib/more_help
1231 file path=usr/lib/newsyslog group=sys mode=0555
1232 file path=usr/lib/passmgmt group=sys mode=0555
1233 file path=usr/lib/pci/pcidr mode=0555
1234 file path=usr/lib/pci/pcidr_plugin.so
1235 file path=usr/lib/pfexecd mode=0555
1236 file path=usr/lib/platexec mode=0555
1237 file path=usr/lib/rcm/modules/SUNW_aggr_rcm.so mode=0555
1238 file path=usr/lib/rcm/modules/SUNW_cluster_rcm.so mode=0555
1239 file path=usr/lib/rcm/modules/SUNW_dump_rcm.so mode=0555
1240 file path=usr/lib/rcm/modules/SUNW_filesys_rcm.so mode=0555
1241 file path=usr/lib/rcm/modules/SUNW_ibpart_rcm.so mode=0555
1242 file path=usr/lib/rcm/modules/SUNW_ip_anon_rcm.so mode=0555
1243 file path=usr/lib/rcm/modules/SUNW_ip_rcm.so mode=0555
1244 file path=usr/lib/rcm/modules/SUNW_mpxio_rcm.so mode=0555
1245 file path=usr/lib/rcm/modules/SUNW_network_rcm.so mode=0555
1246 file path=usr/lib/rcm/modules/SUNW_swap_rcm.so mode=0555
1247 $(sparc_ONLY)file path=usr/lib/rcm/modules/SUNW_ttymux_rcm.so mode=0555
1248 file path=usr/lib/rcm/modules/SUNW_vlan_rcm.so mode=0555

```

```

1249 file path=usr/lib/rcm/modules/SUNW_vnic_rcm.so mode=0555
1250 file path=usr/lib/rcm/rcm_daemon mode=0555
1251 file path=usr/lib/reparse/reparsed group=sys mode=0555
1252 file path=usr/lib/saf/listen group=sys mode=0755
1253 file path=usr/lib/saf/nlps_server group=sys mode=0755
1254 file path=usr/lib/saf/sac group=sys mode=0555
1255 file path=usr/lib/saf/ttymon group=sys mode=0555
1256 file path=usr/lib/sysevent/modules/datalink_mod.so
1257 file path=usr/lib/sysevent/modules/devfsadm_mod.so
1258 file path=usr/lib/sysevent/modules/sysevent_conf_mod.so
1259 file path=usr/lib/sysevent/modules/sysevent_reg_mod.so
1260 file path=usr/lib/sysevent/syseventconfd mode=0555
1261 file path=usr/lib/sysevent/syseventd mode=0555
1262 file path=usr/lib/utmp_update mode=4555
1263 file path=usr/lib/utmpd mode=0555
1264 file path=usr/lib/vtdaemon mode=0555
1265 file path=usr/lib/vtinfo mode=0555
1266 file path=usr/lib/vtxlock mode=0555
1267 file path=usr/sadm/bin/puttext mode=0555
1268 file path=usr/sadm/install/miniroot.db group=sys mode=0444
1269 file path=usr/sadm/install/scripts/i.ipsecalgs group=sys mode=0555
1270 file path=usr/sadm/install/scripts/i.kcfcconf group=sys mode=0555
1271 file path=usr/sadm/install/scripts/i.kmfcconf group=sys mode=0555
1272 file path=usr/sadm/install/scripts/i.manifest group=sys mode=0555
1273 file path=usr/sadm/install/scripts/i.pkcs11conf group=sys mode=0555
1274 file path=usr/sadm/install/scripts/i.rbac group=sys mode=0555
1275 file path=usr/sadm/install/scripts/r.ipsecalgs group=sys mode=0555
1276 file path=usr/sadm/install/scripts/r.kcfcconf group=sys mode=0555
1277 file path=usr/sadm/install/scripts/r.kmfcconf group=sys mode=0555
1278 file path=usr/sadm/install/scripts/r.manifest group=sys mode=0555
1279 file path=usr/sadm/install/scripts/r.pkcs11conf group=sys mode=0555
1280 file path=usr/sadm/install/scripts/r.rbac group=sys mode=0555
1281 file path=usr/sadm/updates mode=0444
1282 $(i386_ONLY)file path=usr/sbin/${ARCH32}/add_drv group=sys mode=0555
1283 $(i386_ONLY)file path=usr/sbin/${ARCH32}/modinfo group=sys mode=0555
1284 $(i386_ONLY)file path=usr/sbin/${ARCH32}/modload group=sys mode=0555
1285 $(i386_ONLY)file path=usr/sbin/${ARCH32}/modunload group=sys mode=0555
1286 $(i386_ONLY)file path=usr/sbin/${ARCH32}/pbind group=sys mode=0555
1287 $(i386_ONLY)file path=usr/sbin/${ARCH32}/prtconf group=sys mode=2555
1288 $(i386_ONLY)file path=usr/sbin/${ARCH32}/psrset group=sys mode=0555
1289 $(i386_ONLY)file path=usr/sbin/${ARCH32}/rem_drv group=sys mode=0555
1290 $(i386_ONLY)file path=usr/sbin/${ARCH32}/swap group=sys mode=2555
1291 $(i386_ONLY)file path=usr/sbin/${ARCH32}/sysdef group=sys mode=2555
1292 $(i386_ONLY)file path=usr/sbin/${ARCH32}/update_drv group=sys mode=0555
1293 $(i386_ONLY)file path=usr/sbin/${ARCH32}/whodo mode=4555
1294 file path=usr/sbin/${ARCH64}/add_drv group=sys mode=0555
1295 file path=usr/sbin/${ARCH64}/modinfo group=sys mode=0555
1296 file path=usr/sbin/${ARCH64}/modload group=sys mode=0555
1297 file path=usr/sbin/${ARCH64}/modunload group=sys mode=0555
1298 file path=usr/sbin/${ARCH64}/pbind group=sys mode=0555
1299 file path=usr/sbin/${ARCH64}/prtconf group=sys mode=2555
1300 file path=usr/sbin/${ARCH64}/psrset group=sys mode=0555
1301 file path=usr/sbin/${ARCH64}/rem_drv group=sys mode=0555
1302 file path=usr/sbin/${ARCH64}/swap group=sys mode=2555
1303 file path=usr/sbin/${ARCH64}/sysdef group=sys mode=2555
1304 file path=usr/sbin/${ARCH64}/update_drv group=sys mode=0555
1305 file path=usr/sbin/${ARCH64}/whodo mode=4555
1306 file path=usr/sbin/6to4relay mode=0555
1307 file path=usr/sbin/acctadm mode=0555
1308 file path=usr/sbin/allocate mode=4555
1309 file path=usr/sbin/arp mode=0555
1310 file path=usr/sbin/audit mode=0555
1311 file path=usr/sbin/auditconfig mode=0555
1312 file path=usr/sbin/auditd mode=0555
1313 file path=usr/sbin/auditrecord mode=0555
1314 file path=usr/sbin/auditreduce mode=0555

```

```

1315 file path=usr/sbin/auditstat mode=0555
1316 file path=usr/sbin/cfgadm mode=0555
1317 file path=usr/sbin/chroot mode=0555
1318 file path=usr/sbin/clear_locks mode=0555
1319 file path=usr/sbin/clinfo mode=0555
1320 file path=usr/sbin/clri mode=0555
1321 file path=usr/sbin/consadm group=sys mode=0555
1322 file path=usr/sbin/cron group=sys mode=0555
1323 file path=usr/sbin/devfsadm group=sys mode=0755
1324 file path=usr/sbin/devinfo mode=0555
1325 file path=usr/sbin/df mode=0555
1326 file path=usr/sbin/dfmounts mode=0555
1327 file path=usr/sbin/dispadm mode=0555
1328 file path=usr/sbin/dminfo mode=0555
1329 file path=usr/sbin/dumpadm mode=0555
1330 file path=usr/sbin/EEPROM group=sys mode=2555
1331 file path=usr/sbin/ff mode=0555
1332 file path=usr/sbin/fmthard group=sys mode=0555
1333 file path=usr/sbin/format mode=0555
1334 file path=usr/sbin/fsck mode=0555
1335 file path=usr/sbin/fstyp group=sys mode=0555
1336 file path=usr/sbin/fuser mode=0555
1337 file path=usr/sbin/getdevpolicy group=sys mode=0555
1338 file path=usr/sbin/getmajor group=sys mode=0755
1339 file path=usr/sbin/groupadd group=sys mode=0555
1340 file path=usr/sbin/groupdel group=sys mode=0555
1341 file path=usr/sbin/groupmod group=sys mode=0555
1342 file path=usr/sbin/grpck mode=0555
1343 file path=usr/sbin/halt mode=0755
1344 file path=usr/sbin/hotplug mode=0555
1345 file path=usr/sbin/idmap mode=0555
1346 file path=usr/sbin/if_mpadm mode=0555
1347 file path=usr/sbin/ikeadm mode=0555
1348 file path=usr/sbin/ikecert mode=0555
1349 file path=usr/sbin/inetadm mode=0555
1350 file path=usr/sbin/inetconv mode=0555
1351 file path=usr/sbin/install mode=0555
1352 file path=usr/sbin/installboot group=sys mode=0555
1353 file path=usr/sbin/ipaddrsel mode=0555
1354 file path=usr/sbin/ipsecalgs mode=0555
1355 file path=usr/sbin/ipsecconf mode=0555
1356 file path=usr/sbin/ipseckey mode=0555
1357 file path=usr/sbin/keyserv group=sys mode=0555
1358 file path=usr/sbin/killall mode=0555
1359 file path=usr/sbin/ksslcfg mode=0555
1360 file path=usr/sbin/link mode=0555
1361 file path=usr/sbin/locator mode=0555
1362 file path=usr/sbin/lofiadm mode=0555
1363 file path=usr/sbin/logadm mode=0555
1364 file path=usr/sbin/makedbm mode=0555
1365 file path=usr/sbin/mkdevalloc mode=0555
1366 file path=usr/sbin/mkfile mode=0555
1367 file path=usr/sbin/mknod mode=0555
1368 file path=usr/sbin/mountall group=sys mode=0555
1369 file path=usr/sbin/msgid mode=0555
1370 file path=usr/sbin/mvdir mode=0555
1371 file path=usr/sbin/ndd mode=0555
1372 file path=usr/sbin/nlsadmin group=adm mode=0755
1373 file path=usr/sbin/nscd mode=0555
1374 file path=usr/sbin/nwamadm mode=0555
1375 file path=usr/sbin/nwamcfg mode=0555
1376 file path=usr/sbin/pmadm group=sys mode=0555
1377 file path=usr/sbin/praudit mode=0555
1378 $(i386_ONLY)file path=usr/sbin/prtdiag group=sys mode=2755
1379 file path=usr/sbin/prtvtoc group=sys mode=0555
1380 file path=usr/sbin/psradm group=sys mode=0555

```

```

1381 file path=usr/sbin/psrinfo group=sys mode=0555
1382 file path=usr/sbin/pwck mode=0555
1383 file path=usr/sbin/pwconv group=sys mode=0555
1384 file path=usr/sbin/raidctl mode=0555
1385 file path=usr/sbin/ramdiskadm mode=0555
1386 file path=usr/sbin/rctladm mode=0555
1387 file path=usr/sbin/root_archive group=sys mode=0555
1388 file path=usr/sbin/rpcbind mode=0555
1389 $(i386_ONLY)file path=usr/sbin/rtc mode=0555
1390 file path=usr/sbin/sacadm group=sys mode=4755
1391 file path=usr/sbin/setmnt mode=0555
1392 file path=usr/sbin/shareall mode=0555
1393 file path=usr/sbin/sharectl mode=0555
1394 file path=usr/sbin/sharemgr mode=0555
1395 file path=usr/sbin/shutdown group=sys mode=0755
1396 file path=usr/sbin/smbios mode=0555
1397 file path=usr/sbin/stmsboot mode=0555
1398 file path=usr/sbin/strace group=sys mode=0555
1399 file path=usr/sbin/strclean group=sys mode=0555
1400 file path=usr/sbin/strerr group=sys mode=0555
1401 file path=usr/sbin/ttydefs group=sys mode=0755
1402 file path=usr/sbin/svcadm mode=0555
1403 file path=usr/sbin/svccfg mode=0555
1404 file path=usr/sbin/syncinit mode=0555
1405 file path=usr/sbin/syncloop mode=0555
1406 file path=usr/sbin/syncstat mode=0555
1407 file path=usr/sbin/syseventadm group=sys mode=0555
1408 file path=usr/sbin/syslogd group=sys mode=0555
1409 file path=usr/sbin/tar mode=0555
1410 file path=usr/sbin/traceroute mode=4555
1411 file path=usr/sbin/trapstat mode=0555
1412 file path=usr/sbin/ttyadm group=sys mode=0755
1413 $(i386_ONLY)file path=usr/sbin/ucodeadm mode=0555
1414 file path=usr/sbin/umountall group=sys mode=0555
1415 file path=usr/sbin/unlink mode=0555
1416 file path=usr/sbin/unshareall mode=0555
1417 file path=usr/sbin/useradd group=sys mode=0555
1418 file path=usr/sbin/userdel group=sys mode=0555
1419 file path=usr/sbin/usermod group=sys mode=0555
1420 $(sparc_ONLY)file path=usr/sbin/virtinfo mode=0555
1421 file path=usr/sbin/volcopy mode=0555
1422 file path=usr/sbin/wall group=tty mode=2555
1423 file path=usr/sbin/zdump mode=0555
1424 file path=usr/sbin/zic mode=0555
1425 file path=usr/share/doc/ksh/COMPATIBILITY
1426 file path=usr/share/doc/ksh/DESIGN
1427 file path=usr/share/doc/ksh/OBSOLETE
1428 file path=usr/share/doc/ksh/README
1429 file path=usr/share/doc/ksh/RELEASE
1430 file path=usr/share/doc/ksh/TYPES
1431 file path=usr/share/doc/ksh/images/callouts/1.png
1432 file path=usr/share/doc/ksh/images/callouts/10.png
1433 file path=usr/share/doc/ksh/images/callouts/2.png
1434 file path=usr/share/doc/ksh/images/callouts/3.png
1435 file path=usr/share/doc/ksh/images/callouts/4.png
1436 file path=usr/share/doc/ksh/images/callouts/5.png
1437 file path=usr/share/doc/ksh/images/callouts/6.png
1438 file path=usr/share/doc/ksh/images/callouts/7.png
1439 file path=usr/share/doc/ksh/images/callouts/8.png
1440 file path=usr/share/doc/ksh/images/callouts/9.png
1441 file path=usr/share/doc/ksh/images/tag_bourne.png
1442 file path=usr/share/doc/ksh/images/tag_i18n.png
1443 file path=usr/share/doc/ksh/images/tag_ksh.png
1444 file path=usr/share/doc/ksh/images/tag_ksh88.png
1445 file path=usr/share/doc/ksh/images/tag_ksh93.png
1446 file path=usr/share/doc/ksh/images/tag_l10n.png

```

```

1447 file path=usr/share/doc/ksh/images/tag_perf.png
1448 file path=usr/share/doc/ksh/shell_styleguide.docbook
1449 file path=usr/share/doc/ksh/shell_styleguide.html
1450 file path=usr/share/lib/mailx/mailx.help
1451 file path=usr/share/lib/mailx/mailx.help.~
1452 file path=usr/share/lib/tabset/3101
1453 file path=usr/share/lib/tabset/beehive
1454 file path=usr/share/lib/tabset/hds
1455 file path=usr/share/lib/tabset/hds3
1456 file path=usr/share/lib/tabset/std
1457 file path=usr/share/lib/tabset/stdcrt
1458 file path=usr/share/lib/tabset/teleray
1459 file path=usr/share/lib/tabset/vt100
1460 file path=usr/share/lib/tabset/wyse-adds
1461 file path=usr/share/lib/tabset/xerox1720
1462 file path=usr/share/lib/termcap
1463 file path=usr/share/lib/unittab
1464 file path=usr/share/lib/xml/dtd/adt_record.dtd.1
1465 file path=usr/share/lib/xml/dtd/kmfpolicy.dtd
1466 file path=usr/share/lib/xml/dtd/service_bundle.dtd.1 group=sys
1467 file path=usr/share/lib/xml/style/adt_record.xsl.1
1468 file path=var/adm/aculog mode=0600 owner=ucup preserve=true
1469 file path=var/adm/spellhist mode=0666 preserve=true
1470 file path=var/adm/utmpx preserve=true
1471 file path=var/adm/wtmpx group=adm owner=adm preserve=true
1472 file path=var/log/authlog group=sys mode=0600 preserve=true
1473 file path=var/log/syslog group=sys preserve=true
1474 file path=var/sadm/system/admin/default_java group=sys mode=0444
1475 file path=var/saf/zsmon/log group=sys preserve=true
1476 file path=var/spool/cron/crontabs/adm group=sys mode=0600 preserve=true
1477 file path=var/spool/cron/crontabs/root group=sys mode=0600 preserve=true
1478 hardlink path=etc/rc2.d/S20syssetup target=../etc/init.d/syssetup
1479 hardlink path=etc/rc2.d/S73cachefs.daemon \
1480   target=../etc/init.d/cachefs.daemon
1481 hardlink path=etc/rc2.d/S89PRESERVE target=../etc/init.d/PRESERVE
1482 $(sparc_ONLY)hardlink path=etc/svc/profile/platform_SUNW,Sun-Fire-V890.xml \
1483   target=../platform_SUNW,Sun-Fire-880.xml
1484 $(sparc_ONLY)hardlink \
1485   path=etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-40.xml \
1486   target=../platform_SUNW,UltraSPARC-IIi-Netract.xml
1487 $(sparc_ONLY)hardlink \
1488   path=etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-60.xml \
1489   target=../platform_SUNW,UltraSPARC-IIi-Netract.xml
1490 hardlink path=sbin/rc5 target=../sbin/rc0
1491 hardlink path=sbin/rc6 target=../sbin/rc0
1492 hardlink path=usr/bin/$(ARCH32)/encrypt target=decrypt
1493 hardlink path=usr/bin/$(ARCH32)/ksh target=ksh93
1494 hardlink path=usr/bin/$(ARCH32)/mac target=digest
1495 hardlink path=usr/bin/$(ARCH32)/rksh target=ksh93
1496 hardlink path=usr/bin/$(ARCH32)/rksh93 target=ksh93
1497 $(i386_ONLY)hardlink path=usr/bin/$(ARCH32)/w target=uptime
1498 hardlink path=usr/bin/$(ARCH64)/encrypt target=decrypt
1499 hardlink path=usr/bin/$(ARCH64)/ksh target=ksh93
1500 hardlink path=usr/bin/$(ARCH64)/mac target=digest
1501 hardlink path=usr/bin/$(ARCH64)/rksh target=ksh93
1502 hardlink path=usr/bin/$(ARCH64)/rksh93 target=ksh93
1503 hardlink path=usr/bin/$(ARCH64)/w target=uptime
1504 hardlink path=usr/bin/bg target=../usr/bin/alias
1505 hardlink path=usr/bin/cd target=../usr/bin/alias
1506 hardlink path=usr/bin/cksum target=../usr/bin/alias
1507 hardlink path=usr/bin/cmp target=../usr/bin/alias
1508 hardlink path=usr/bin/comm target=../usr/bin/alias
1509 hardlink path=usr/bin/command target=../usr/bin/alias
1510 hardlink path=usr/bin/cut target=../usr/bin/alias
1511 hardlink path=usr/bin/decrypt target=../usr/lib/isaexec
1512 hardlink path=usr/bin/digest target=../usr/lib/isaexec

```

```

1513 hardlink path=usr/bin/dispgid target=../usr/bin/ckgid
1514 hardlink path=usr/bin/dispuid target=../usr/bin/ckuid
1515 hardlink path=usr/bin/edit target=../usr/bin/edit
1516 hardlink path=usr/bin/encrypt target=../usr/lib/isaexec
1517 hardlink path=usr/bin/fc target=../usr/bin/alias
1518 hardlink path=usr/bin/fg target=../usr/bin/alias
1519 hardlink path=usr/bin/getopts target=../usr/bin/alias
1520 hardlink path=usr/bin/hash target=../usr/bin/alias
1521 hardlink path=usr/bin/i386 target=../usr/bin/i286
1522 hardlink path=usr/bin/i486 target=../usr/bin/i286
1523 hardlink path=usr/bin/i860 target=../usr/bin/i286
1524 hardlink path=usr/bin/i86pc target=../usr/bin/i286
1525 hardlink path=usr/bin/iAPX286 target=../usr/bin/i286
1526 hardlink path=usr/bin/jobs target=../usr/bin/alias
1527 hardlink path=usr/bin/join target=../usr/bin/alias
1528 hardlink path=usr/bin/kill target=../usr/bin/alias
1529 hardlink path=usr/bin/ksh target=../usr/lib/isaexec
1530 hardlink path=usr/bin/ksh93 target=../usr/lib/isaexec
1531 hardlink path=usr/bin/ln target=../usr/bin/cp
1532 hardlink path=usr/bin/logname target=../usr/bin/alias
1533 hardlink path=usr/bin/m68k target=../usr/bin/i286
1534 hardlink path=usr/bin/mac target=../usr/lib/isaexec
1535 hardlink path=usr/bin/mc68000 target=../usr/bin/i286
1536 hardlink path=usr/bin/mc68010 target=../usr/bin/i286
1537 hardlink path=usr/bin/mc68020 target=../usr/bin/i286
1538 hardlink path=usr/bin/mc68030 target=../usr/bin/i286
1539 hardlink path=usr/bin/mc68040 target=../usr/bin/i286
1540 hardlink path=usr/bin/mv target=../usr/bin/cp
1541 hardlink path=usr/bin/newtask target=../usr/lib/isaexec
1542 hardlink path=usr/bin/nohup target=../usr/lib/isaexec
1543 hardlink path=usr/bin/page target=../usr/bin/more
1544 hardlink path=usr/bin/paste target=../usr/bin/alias
1545 hardlink path=usr/bin/pdpl1 target=../usr/bin/i286
1546 hardlink path=usr/bin/pfbash target=../usr/bin/pfexec
1547 hardlink path=usr/bin/pfcsh target=../usr/bin/pfexec
1548 hardlink path=usr/bin/pfksh target=../usr/bin/pfexec
1549 hardlink path=usr/bin/pfksh93 target=../usr/bin/pfexec
1550 hardlink path=usr/bin/pfrksh target=../usr/bin/pfexec
1551 hardlink path=usr/bin/pfrksh93 target=../usr/bin/pfexec
1552 hardlink path=usr/bin/pfsh target=../usr/bin/pfexec
1553 hardlink path=usr/bin/pftcsh target=../usr/bin/pfexec
1554 hardlink path=usr/bin/pfzsh target=../usr/bin/pfexec
1555 hardlink path=usr/bin/pkill target=../usr/bin/pgrep
1556 hardlink path=usr/bin/prctl target=../usr/lib/isaexec
1557 hardlink path=usr/bin/print target=../usr/bin/alias
1558 hardlink path=usr/bin/prstat target=../usr/lib/isaexec
1559 hardlink path=usr/bin/ps target=../usr/lib/isaexec
1560 hardlink path=usr/bin/read target=../usr/bin/alias
1561 hardlink path=usr/bin/red target=../usr/bin/ed
1562 hardlink path=usr/bin/rev target=../usr/bin/alias
1563 hardlink path=usr/bin/rksh target=../usr/lib/isaexec
1564 hardlink path=usr/bin/rksh93 target=../usr/lib/isaexec
1565 hardlink path=usr/bin/savecore target=../usr/lib/isaexec
1566 hardlink path=usr/bin/setuname target=../usr/lib/isaexec
1567 hardlink path=usr/bin/sleep target=../usr/bin/alias
1568 hardlink path=usr/bin/sparc target=../usr/bin/i286
1569 hardlink path=usr/bin/sum target=../usr/bin/alias
1570 hardlink path=usr/bin/sun target=../usr/bin/i286
1571 hardlink path=usr/bin/sun2 target=../usr/bin/i286
1572 hardlink path=usr/bin/sun3 target=../usr/bin/i286
1573 hardlink path=usr/bin/sun3x target=../usr/bin/i286
1574 hardlink path=usr/bin/sun4 target=../usr/bin/i286
1575 hardlink path=usr/bin/sun4c target=../usr/bin/i286
1576 hardlink path=usr/bin/sun4d target=../usr/bin/i286
1577 hardlink path=usr/bin/sun4e target=../usr/bin/i286
1578 hardlink path=usr/bin/sun4m target=../usr/bin/i286

```



```

1579 hardlink path=usr/bin/tee target=../usr/bin/alias
1580 hardlink path=usr/bin/test target=../usr/bin/alias
1581 hardlink path=usr/bin/touch target=../usr/bin/settime
1582 hardlink path=usr/bin/type target=../usr/bin/alias
1583 hardlink path=usr/bin/u370 target=../usr/bin/i286
1584 hardlink path=usr/bin/u3b target=../usr/bin/i286
1585 hardlink path=usr/bin/u3b15 target=../usr/bin/i286
1586 hardlink path=usr/bin/u3b2 target=../usr/bin/i286
1587 hardlink path=usr/bin/u3b5 target=../usr/bin/i286
1588 hardlink path=usr/bin/ulimit target=../usr/bin/alias
1589 hardlink path=usr/bin/umask target=../usr/bin/alias
1590 hardlink path=usr/bin/unalias target=../usr/bin/alias
1591 hardlink path=usr/bin/uniq target=../usr/bin/alias
1592 hardlink path=usr/bin/uptime target=../usr/lib/isaexec
1593 hardlink path=usr/bin/vax target=../usr/bin/i286
1594 hardlink path=usr/bin/vedit target=../usr/bin/edit
1595 hardlink path=usr/bin/w target=../usr/lib/isaexec
1596 hardlink path=usr/bin/wait target=../usr/bin/alias
1597 hardlink path=usr/bin/wc target=../usr/bin/alias
1598 hardlink path=usr/has/bin/ex target=edit
1599 hardlink path=usr/has/bin/pfsh target=../usr/bin/pfexec
1600 hardlink path=usr/has/bin/vedit target=edit
1601 hardlink path=usr/has/bin/vi target=edit
1602 hardlink path=usr/has/bin/view target=edit
1603 hardlink path=usr/lib/fs/ufs/hfs/fstyp target=../usr/sbin/fstyp
1604 hardlink path=usr/lib/fs/ufs/dcopy target=../usr/lib/fs/ufs/clri
1605 hardlink path=usr/lib/fs/ufs/fstyp target=../usr/sbin/fstyp
1606 hardlink path=usr/lib/fs/ufs/quotan \
1607   target=../usr/lib/fs/ufs/quotaoiff
1608 hardlink path=usr/lib/inet/in.iked target=../usr/lib/isaexec
1609 hardlink path=usr/sadm/bin/dispgid target=../usr/bin/ckgid
1610 hardlink path=usr/sadm/bin/dispuid target=../usr/bin/ckuid
1611 hardlink path=usr/sadm/bin/errange target=../usr/bin/ckrange
1612 hardlink path=usr/sadm/bin/errdate target=../usr/bin/ckdate
1613 hardlink path=usr/sadm/bin/errgid target=../usr/bin/ckgid
1614 hardlink path=usr/sadm/bin/errint target=../usr/bin/ckint
1615 hardlink path=usr/sadm/bin/erritem target=../usr/bin/ckitem
1616 hardlink path=usr/sadm/bin/errpath target=../usr/bin/ckpath
1617 hardlink path=usr/sadm/bin/errstr target=../usr/bin/ckstr
1618 hardlink path=usr/sadm/bin/errtime target=../usr/bin/cktime
1619 hardlink path=usr/sadm/bin/erruid target=../usr/bin/ckuid
1620 hardlink path=usr/sadm/bin/erryorn target=../usr/bin/ckyorn
1621 hardlink path=usr/sadm/bin/helpdate target=../usr/bin/ckdate
1622 hardlink path=usr/sadm/bin/helpgid target=../usr/bin/ckgid
1623 hardlink path=usr/sadm/bin/helpint target=../usr/bin/ckint
1624 hardlink path=usr/sadm/bin/helpitem target=../usr/bin/ckitem
1625 hardlink path=usr/sadm/bin/helppath target=../usr/bin/ckpath
1626 hardlink path=usr/sadm/bin/helpprange target=../usr/bin/ckrange
1627 hardlink path=usr/sadm/bin/helpstr target=../usr/bin/ckstr
1628 hardlink path=usr/sadm/bin/helptime target=../usr/bin/cktime
1629 hardlink path=usr/sadm/bin/helpuid target=../usr/bin/ckuid
1630 hardlink path=usr/sadm/bin/helpyorn target=../usr/bin/ckyorn
1631 hardlink path=usr/sadm/bin/valdate target=../usr/bin/ckdate
1632 hardlink path=usr/sadm/bin/valgid target=../usr/bin/ckgid
1633 hardlink path=usr/sadm/bin/valint target=../usr/bin/ckint
1634 hardlink path=usr/sadm/bin/valpath target=../usr/bin/ckpath
1635 hardlink path=usr/sadm/bin/valrange target=../usr/bin/ckrange
1636 hardlink path=usr/sadm/bin/valstr target=../usr/bin/ckstr
1637 hardlink path=usr/sadm/bin/valtime target=../usr/bin/cktime
1638 hardlink path=usr/sadm/bin/valuid target=../usr/bin/ckuid
1639 hardlink path=usr/sadm/bin/valyorn target=../usr/bin/ckyorn
1640 hardlink path=usr/sbin/add_drv target=../usr/lib/isaexec
1641 hardlink path=usr/sbin/audlinks target=../devfsadm
1642 hardlink path=usr/sbin/consadm target=../usr/sbin/consadm
1643 hardlink path=usr/sbin/deallocate target=../usr/sbin/allocate
1644 hardlink path=usr/sbin/devlinks target=../devfsadm

```

```

1645 hardlink path=usr/sbin/dfshares target=../usr/sbin/dfmounts
1646 hardlink path=usr/sbin/disks target=../devfsadm
1647 hardlink path=usr/sbin/drvconfig target=../devfsadm
1648 hardlink path=usr/sbin/list_devices target=../usr/sbin/allocate
1649 hardlink path=usr/sbin/mkdevmaps target=../usr/sbin/mkdevalloc
1650 hardlink path=usr/sbin/modinfo target=../usr/lib/isaexec
1651 hardlink path=usr/sbin/modload target=../usr/lib/isaexec
1652 hardlink path=usr/sbin/modunload target=../usr/lib/isaexec
1653 hardlink path=usr/sbin/pbind target=../usr/lib/isaexec
1654 hardlink path=usr/sbin/ports target=../devfsadm
1655 hardlink path=usr/sbin/poweroff target=../halt
1656 hardlink path=usr/sbin/prtconf target=../usr/lib/isaexec
1657 $(sparc_ONLY)hardlink path=usr/sbin/prtdiag target=../usr/lib/platexec
1658 hardlink path=usr/sbin/psrset target=../usr/lib/isaexec
1659 hardlink path=usr/sbin/reboot target=../halt
1660 hardlink path=usr/sbin/rem_drv target=../usr/lib/isaexec
1661 hardlink path=usr/sbin/roleadd target=../usr/sbin/useradd
1662 hardlink path=usr/sbin/roledel target=../usr/sbin/userdel
1663 hardlink path=usr/sbin/rolemod target=../usr/sbin/usermod
1664 hardlink path=usr/sbin/share target=../usr/sbin/sharemgr
1665 hardlink path=usr/sbin/swap target=../usr/lib/isaexec
1666 hardlink path=usr/sbin/sysdef target=../usr/lib/isaexec
1667 hardlink path=usr/sbin/tapes target=../devfsadm
1668 hardlink path=usr/sbin/unshare target=../usr/sbin/sharemgr
1669 hardlink path=usr/sbin/update_drv target=../usr/lib/isaexec
1670 hardlink path=usr/sbin/whodo target=../usr/lib/isaexec
1671 legacy pkg=SUNWcsr \
1672   desc="core software for a specific instruction-set architecture" \
1673   name="Core Solaris, (Root)"
1674 legacy pkg=SUNWcsu \
1675   desc="core software for a specific instruction-set architecture" \
1676   name="Core Solaris, (Usr)"
1677 legacy pkg=SUNWftpr desc="FTP Server Configuration Files" \
1678   name="FTP Server, (Root)"
1679 license cr_Sun license=cr_Sun
1680 license lic_CDDL license=lic_CDDL
1681 license usr/src/cmd/cmd-inet/sbin/ifparse/THIRDPARTYLICENSE \
1682   license=usr/src/cmd/cmd-inet/sbin/ifparse/THIRDPARTYLICENSE
1683 license usr/src/cmd/cmd-inet/usr.lib/in.mpathd/THIRDPARTYLICENSE \
1684   license=usr/src/cmd/cmd-inet/usr.lib/in.mpathd/THIRDPARTYLICENSE
1685 license usr/src/cmd/cmd-inet/usr/sbin/THIRDPARTYLICENSE.arp \
1686   license=usr/src/cmd/cmd-inet/usr/sbin/THIRDPARTYLICENSE.arp
1687 license usr/src/cmd/cmd-inet/usr/sbin/THIRDPARTYLICENSE.route \
1688   license=usr/src/cmd/cmd-inet/usr/sbin/THIRDPARTYLICENSE.route
1689 license usr/src/cmd/cmd-inet/usr/sbin/ifconfig/THIRDPARTYLICENSE \
1690   license=usr/src/cmd/cmd-inet/usr/sbin/ifconfig/THIRDPARTYLICENSE
1691 license usr/src/cmd/cmd-inet/usr/sbin/in.ftpd/LICENSE \
1692   license=usr/src/cmd/cmd-inet/usr/sbin/in.ftpd/LICENSE
1693 license usr/src/cmd/cmd-inet/usr/sbin/traceroute/THIRDPARTYLICENSE \
1694   license=usr/src/cmd/cmd-inet/usr/sbin/traceroute/THIRDPARTYLICENSE
1695 license usr/src/cmd/cron/THIRDPARTYLICENSE \
1696   license=usr/src/cmd/cron/THIRDPARTYLICENSE
1697 license usr/src/cmd/csh/THIRDPARTYLICENSE \
1698   license=usr/src/cmd/csh/THIRDPARTYLICENSE
1699 license usr/src/cmd/eeprom/THIRDPARTYLICENSE \
1700   license=usr/src/cmd/eeprom/THIRDPARTYLICENSE
1701 license usr/src/cmd/fs.d/ufs/THIRDPARTYLICENSE \
1702   license=usr/src/cmd/fs.d/ufs/THIRDPARTYLICENSE
1703 license usr/src/cmd/mt/THIRDPARTYLICENSE \
1704   license=usr/src/cmd/mt/THIRDPARTYLICENSE
1705 license usr/src/cmd/script/THIRDPARTYLICENSE \
1706   license=usr/src/cmd/script/THIRDPARTYLICENSE
1707 license usr/src/cmd/sed/THIRDPARTYLICENSE \
1708   license=usr/src/cmd/sed/THIRDPARTYLICENSE
1709 license usr/src/cmd/stat/vmstat/THIRDPARTYLICENSE \
1710   license=usr/src/cmd/stat/vmstat/THIRDPARTYLICENSE

```

```

1711 license usr/src/cmd/tail/THIRDPARTYLICENSE \
1712     license=usr/src/cmd/tail/THIRDPARTYLICENSE
1713 license usr/src/cmd/tip/THIRDPARTYLICENSE \
1714     license=usr/src/cmd/tip/THIRDPARTYLICENSE
1715 license usr/src/cmd/tr/THIRDPARTYLICENSE \
1716     license=usr/src/cmd/tr/THIRDPARTYLICENSE
1717 license usr/src/cmd/vi/THIRDPARTYLICENSE \
1718     license=usr/src/cmd/vi/THIRDPARTYLICENSE
1719 license usr/src/cmd/which/THIRDPARTYLICENSE \
1720     license=usr/src/cmd/which/THIRDPARTYLICENSE
1721 license usr/src/cmd/xstr/THIRDPARTYLICENSE \
1722     license=usr/src/cmd/xstr/THIRDPARTYLICENSE
1723 license usr/src/common/bzip2/LICENSE license=usr/src/common/bzip2/LICENSE
1724 link path=bin target=../usr/bin
1725 link path=etc/TIMEZONE target=../default/init
1726 link path=etc/autopush target=../sbin/autopush
1727 link path=etc/cfgadm target=../usr/sbin/cfgadm
1728 link path=etc/clri target=../usr/sbin/clri
1729 link path=etc/cron target=../usr/sbin/cron
1730 link path=etc/dcopy target=../usr/sbin/dcopy
1731 link path=etc/ff target=../usr/sbin/ff
1732 link path=etc/fmthard target=../usr/sbin/fmthard
1733 link path=etc/format target=../usr/sbin/format
1734 link path=etc/fsck target=../usr/sbin/fsck
1735 link path=etc/fsdb target=../usr/sbin/fsdb
1736 link path=etc/fstyp target=../usr/sbin/fstyp
1737 link path=etc/getty target=../usr/lib/saf/ttymon
1738 link path=etc/grpck target=../usr/sbin/grpck
1739 link path=etc/halt target=../usr/sbin/halt
1740 link path=etc/hosts target=../inet/hosts
1741 link path=etc/inet/ipnodes target=../hosts
1742 link path=etc/inetd.conf target=../inet/inetd.conf
1743 link path=etc/init target=../sbin/init
1744 link path=etc/install target=../usr/sbin/install
1745 link path=etc/killall target=../usr/sbin/killall
1746 link path=etc/labelit target=../usr/sbin/labelit
1747 link path=etc/lib/ld.so.1 target=../lib/ld.so.1
1748 link path=etc/lib/libdl.so.1 target=../lib/libdl.so.1
1749 link path=etc/lib/nss_files.so.1 target=../lib/nss_files.so.1
1750 link path=etc/log target=../var/adm/log
1751 link path=etc/mkfs target=../usr/sbin/mkfs
1752 link path=etc/mknod target=../usr/sbin/mknod
1753 link path=etc/mount target=../sbin/mount
1754 link path=etc/mountall target=../sbin/mountall
1755 link path=etc/ncheck target=../usr/sbin/ncheck
1756 link path=etc/netmasks target=../inet/netmasks
1757 link path=etc/networks target=../inet/networks
1758 link path=etc/protocols target=../inet/protocols
1759 link path=etc/prtconf target=../usr/sbin/prtconf
1760 link path=etc/prtvtoc target=../usr/sbin/prtvtoc
1761 link path=etc/rc0 target=../sbin/rc0
1762 link path=etc/rc1 target=../sbin/rc1
1763 link path=etc/rc2 target=../sbin/rc2
1764 link path=etc/rc3 target=../sbin/rc3
1765 link path=etc/rc5 target=../sbin/rc5
1766 link path=etc/rc6 target=../sbin/rc6
1767 link path=etc/rcS target=../sbin/rcS
1768 link path=etc/reboot target=../usr/sbin/halt
1769 link path=etc/security/audit/localhost/files target=../var/audit
1770 link path=etc/services target=../inet/services
1771 link path=etc/setmnt target=../usr/sbin/setmnt
1772 link path=etc/shutdown target=../usr/sbin/shutdown
1773 link path=etc/sulogin target=../sbin/sulogin
1774 link path=etc/swap target=../usr/sbin/swap
1775 link path=etc/swapadd target=../sbin/swapadd
1776 link path=etc/sysdef target=../usr/sbin/sysdef

```

```

1777 link path=etc/tar target=../usr/sbin/tar
1778 link path=etc/telinit target=../sbin/init
1779 link path=etc/uadmin target=../sbin/uadmin
1780 link path=etc/umount target=../sbin/umount
1781 link path=etc/umountall target=../sbin/umountall
1782 link path=etc/utmpx target=../var/adm/utmpx
1783 link path=etc/volcopy target=../usr/sbin/volcopy
1784 link path=etc/wall target=../usr/sbin/wall
1785 link path=etc/whodo target=../usr/sbin/whodo
1786 link path=etc/wtmpx target=../var/adm/wtmpx
1787 link path=sbin/in.mpathd target=../lib/inet/in.mpathd
1788 link path=sbin/jsh target=../usr/bin/ksh93
1789 link path=sbin/pfsh target=../usr/bin/pfexec
1790 link path=sbin/sh target=../usr/bin/$(ARCH32)/ksh93
1791 link path=sbin/su target=../usr/bin/su
1792 link path=usr/adm target=../var/adm
1793 link path=usr/bin/cachefspack target=../lib/fs/cachefs/cachefspack
1794 link path=usr/bin/cachefsstat target=../lib/fs/cachefs/cachefsstat
1795 link path=usr/bin/df target=../sbin/df
1796 link path=usr/bin/jsh target=ksh93
1797 link path=usr/bin/pwconv target=../sbin/pwconv
1798 link path=usr/bin/rmail target=../mail
1799 link path=usr/bin/sh target=$(ARCH32)/ksh93
1800 link path=usr/bin/strclean target=../sbin/strclean
1801 link path=usr/bin/strerr target=../sbin/strerr
1802 link path=usr/bin/sync target=../sbin/sync
1803 link path=usr/bin/tar target=../sbin/tar
1804 link path=usr/bin/uname target=../sbin/uname
1805 link path=usr/ccs/bin/m4 target=../bin/m4
1806 link path=usr/has/bin/jsh target=sh
1807 link path=usr/has/lib/rsh target=../bin/sh
1808 link path=usr/lib/$(ARCH64)/ld.so.1 target=../lib/$(ARCH64)/ld.so.1
1809 link path=usr/lib/cron target=../etc/cron.d
1810 link path=usr/lib/devfsadm/devfsadmd target=../sbin/devfsadm
1811 link path=usr/lib/embedded_su target=../bin/su
1812 link path=usr/lib/fs/dev/mount target=../etc/fs/dev/mount
1813 link path=usr/lib/fs/hsfs/mount target=../etc/fs/hsfs/mount
1814 link path=usr/lib/fs/ufs/mount target=../etc/fs/ufs/mount
1815 link path=usr/lib/inet/in.mpathd target=../lib/inet/in.mpathd
1816 link path=usr/lib/ld.so.1 target=../lib/ld.so.1
1817 link path=usr/lib/locale/POSIX target=../C
1818 link path=usr/lib/rsh target=../bin/ksh93
1819 link path=usr/lib/secure/32 target=..
1820 link path=usr/lib/secure/64 target=$(ARCH64)
1821 link path=usr/lib/wusbcd target=../sbin/wusbadm
1822 link path=usr/mail target=../var/mail
1823 link path=usr/net/nls/listen target=../lib/saf/listen
1824 link path=usr/net/nls/nlps_server target=../lib/saf/nlps_server
1825 link path=usr/news target=../var/news
1826 link path=usr/preserve target=../var/preserve
1827 link path=usr/pub target=../share/lib/pub
1828 link path=usr/sbin/autopush target=../sbin/autopush
1829 link path=usr/sbin/bootadm target=../sbin/bootadm
1830 link path=usr/sbin/cachefslog target=../lib/fs/cachefs/cachefslog
1831 link path=usr/sbin/cachefswssize target=../lib/fs/cachefs/cachefswssize
1832 link path=usr/sbin/cfsadmin target=../lib/fs/cachefs/cfsadmin
1833 link path=usr/sbin/cryptoadm target=../sbin/cryptoadm
1834 link path=usr/sbin/dcopy target=../clri
1835 link path=usr/sbin/devnm target=../df
1836 link path=usr/sbin/dladm target=../sbin/dladm
1837 link path=usr/sbin/dlstat target=../sbin/dlstat
1838 link path=usr/sbin/edquota target=../lib/fs/ufs/edquota
1839 link path=usr/sbin/fdisk target=../sbin/fdisk
1840 link path=usr/sbin/fiocompress target=../sbin/fiocompress
1841 link path=usr/sbin/flowadm target=../sbin/flowadm
1842 link path=usr/sbin/flowstat target=../sbin/flowstat

```

```
1843 link path=usr/sbin/fsdb target=./clri
1844 link path=usr/sbin/fsirand target=./lib/fs/ufs/fsirand
1845 link path=usr/sbin/fssnap target=./clri
1846 link path=usr/sbin/hostconfig target=./../sbin/hostconfig
1847 link path=usr/sbin/ifconfig target=./../sbin/ifconfig
1848 link path=usr/sbin/inetd target=./lib/inet/inetd
1849 link path=usr/sbin/init target=./../sbin/init
1850 $(i386_ONLY)link path=usr/sbin/installgrub target=./../sbin/installgrub
1851 link path=usr/sbin/ipadm target=./../sbin/ipadm
1852 link path=usr/sbin/impstat target=./../sbin/impstat
1853 link path=usr/sbin/labelit target=./clri
1854 link path=usr/sbin/lockfs target=./lib/fs/ufs/lockfs
1855 link path=usr/sbin/mkfs target=./clri
1856 link path=usr/sbin/mount target=./../sbin/mount
1857 link path=usr/sbin/ncheck target=./ff
1858 link path=usr/sbin/newfs target=./lib/fs/ufs/newfs
1859 link path=usr/sbin/quot target=./lib/fs/ufs/quot
1860 link path=usr/sbin/quota target=./lib/fs/ufs/quota
1861 link path=usr/sbin/quotacheck target=./lib/fs/ufs/quotacheck
1862 link path=usr/sbin/quotaoff target=./lib/fs/ufs/quotaoff
1863 link path=usr/sbin/quotaon target=./lib/fs/ufs/quotaon
1864 link path=usr/sbin/repquota target=./lib/fs/ufs/repquota
1865 link path=usr/sbin/route target=./../sbin/route
1866 link path=usr/sbin/routeadm target=./../sbin/routeadm
1867 link path=usr/sbin/sync target=./../sbin/sync
1868 link path=usr/sbin/tunefs target=./lib/fs/ufs/tunefs
1869 link path=usr/sbin/tzreload target=./../sbin/tzreload
1870 link path=usr/sbin/uadmin target=./../sbin/uadmin
1871 link path=usr/sbin/ufsdump target=./lib/fs/ufs/ufsdump
1872 link path=usr/sbin/ufsrestore target=./lib/fs/ufs/ufsrestore
1873 link path=usr/sbin/umount target=./../sbin/umount
1874 link path=usr/sbin/wusbadm target=./../sbin/wusbadm
1875 link path=usr/spool target=./var/spool
1876 link path=usr/src target=./share/src
1877 link path=usr/tmp target=./var/tmp
1878 link path=var/ld/32 target=.
1879 link path=var/ld/64 target=$(ARCH64)
1880 #
1881 # The bootadm binary needs the etc/release file.
1882 #
1883 depend fmri=release/name type=require
1884 #
1885 # intrd and others use the illumos-defaulted perl interpreter
1886 #
1887 depend fmri=runtime/perl-510 type=require
1888 #
1889 # The loadkeys binary needs the keytables.
1890 #
1891 depend fmri=system/data/keyboard/keytables type=require
1892 #
1893 # Depend on terminfo data.
1894 #
1895 depend fmri=system/data/terminfo type=require
1896 #
1897 # Depend on zoneinfo data.
1898 #
1899 depend fmri=system/data/zoneinfo type=require
```

```

*****
60001 Sat Jun 1 10:02:52 2013
new/usr/src/pkg/manifests/system-library.mf
2989 Eliminate use of LOGNAME_MAX in ON
1166 useradd have warning with name more 8 chars
*****
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 #
23 # Copyright 2011 Nexenta Systems, Inc. All rights reserved.
24 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
25 # Copyright 2012 OmniTI Computer Consulting, Inc. All rights reserved.
26 # Copyright (c) 2013 Gary Mills
27 #
28 #
29 <include system-library.man3.inc>
30 <include system-library.man3bsm.inc>
31 <include system-library.man3c.inc>
32 <include system-library.man3c_db.inc>
33 <include system-library.man3cfgadm.inc>
34 <include system-library.man3computil.inc>
35 <include system-library.man3contract.inc>
36 <include system-library.man3curses.inc>
37 <include system-library.man3devid.inc>
38 <include system-library.man3devinfo.inc>
39 <include system-library.man3dlpi.inc>
40 <include system-library.man3elf.inc>
41 <include system-library.man3exacct.inc>
42 <include system-library.man3ext.inc>
43 <include system-library.man3fstyp.inc>
44 <include system-library.man3gen.inc>
45 <include system-library.man3kstat.inc>
46 <include system-library.man3kvm.inc>
47 <include system-library.man3ldap.inc>
48 <include system-library.man3lgrp.inc>
49 <include system-library.man3lib.inc>
50 <include system-library.man3mail.inc>
51 <include system-library.man3malloc.inc>
52 <include system-library.man3mp.inc>
53 <include system-library.man3nsl.inc>
54 <include system-library.man3nvpair.inc>
55 <include system-library.man3pam.inc>
56 <include system-library.man3scf.inc>
57 <include system-library.man3sec.inc>
58 <include system-library.man3secdb.inc>
59 <include system-library.man3sip.inc>
60 <include system-library.man3socket.inc>

```

```

61 <include system-library.man3tsol.inc>
62 <include system-library.man3uuid.inc>
63 <include system-library.man3volmgt.inc>
64 <include system-library.man3xcurses.inc>
65 <include system-library.man3xnet.inc>
66 <include system-library.man4.inc>
67 <include system-library.man5.inc>
68 <include system-library.man7p.inc>
69 set name=pkg.fmri value=pkg:/system/library@$(PKGVERS)
70 set name=pkg.description \
71     value="core shared libraries for a specific instruction-set architecture"
72 set name=pkg.summary value="Core Solaris, (Shared Libs)"
73 set name=info.classification value=org.opensolaris.category.2008:System/Core
74 set name=variant.arch value=$(ARCH)
75 $(i386_ONLY)dir path=etc group=sys
76 $(i386_ONLY)dir path=etc/flash group=sys
77 $(i386_ONLY)dir path=etc/flash/postcreation group=sys mode=0700
78 $(i386_ONLY)dir path=etc/flash/precreation group=sys mode=0700
79 $(i386_ONLY)dir path=etc/flash/preexit group=sys mode=0700
80 dir path=lib
81 dir path=lib/$(ARCH64)
82 dir path=lib/crypto
83 dir path=lib/crypto/$(ARCH64)
84 dir path=lib/mpxio
85 dir path=lib/secure
86 dir path=lib/secure/$(ARCH64)
87 dir path=usr group=sys
88 dir path=usr/bin
89 dir path=usr/ccs
90 dir path=usr/ccs/lib
91 dir path=usr/ccs/lib/$(ARCH64)
92 dir path=usr/lib
93 dir path=usr/lib/$(ARCH64)
94 dir path=usr/lib/cfgadm
95 dir path=usr/lib/cfgadm/$(ARCH64)
96 dir path=usr/lib/iconv/$(ARCH64)
97 $(i386_ONLY)dir path=usr/lib/libc
98 dir path=usr/lib/lwp
99 dir path=usr/lib/lwp/$(ARCH64)
100 dir path=usr/lib/python2.6
101 dir path=usr/lib/python2.6/vendor-packages
102 dir path=usr/lib/python2.6/vendor-packages/solaris
103 dir path=usr/lib/raidcfg
104 dir path=usr/lib/raidcfg/$(ARCH64)
105 dir path=usr/lib/scsi
106 dir path=usr/lib/scsi/$(ARCH64)
107 dir path=usr/lib/scsi/plugins
108 dir path=usr/lib/scsi/plugins/scsi
109 dir path=usr/lib/scsi/plugins/scsi/engines
110 dir path=usr/lib/scsi/plugins/scsi/engines/$(ARCH64)
111 dir path=usr/lib/scsi/plugins/ses
112 dir path=usr/lib/scsi/plugins/ses/framework
113 dir path=usr/lib/scsi/plugins/ses/framework/$(ARCH64)
114 dir path=usr/lib/scsi/plugins/ses/vendor
115 $(sparc_ONLY)dir path=usr/lib/scsi/plugins/ses/vendor/$(ARCH64)
116 dir path=usr/lib/scsi/plugins/smp
117 dir path=usr/lib/scsi/plugins/smp/engine
118 dir path=usr/lib/scsi/plugins/smp/engine/$(ARCH64)
119 dir path=usr/lib/scsi/plugins/smp/framework
120 dir path=usr/lib/scsi/plugins/smp/framework/$(ARCH64)
121 dir path=usr/lib/security
122 dir path=usr/lib/security/$(ARCH64)
123 dir path=usr/share/man
124 dir path=usr/share/man/man3
125 dir path=usr/share/man/man3bsm
126 dir path=usr/share/man/man3c

```

```

127 dir path=usr/share/man/man3c_db
128 dir path=usr/share/man/man3cfgadm
129 dir path=usr/share/man/man3computil
130 dir path=usr/share/man/man3contract
131 dir path=usr/share/man/man3curses
132 dir path=usr/share/man/man3devid
133 dir path=usr/share/man/man3devinfo
134 dir path=usr/share/man/man3dlpi
135 dir path=usr/share/man/man3elf
136 dir path=usr/share/man/man3exacct
137 dir path=usr/share/man/man3ext
138 dir path=usr/share/man/man3fstyp
139 dir path=usr/share/man/man3gen
140 dir path=usr/share/man/man3kstat
141 dir path=usr/share/man/man3kvm
142 dir path=usr/share/man/man3ldap
143 dir path=usr/share/man/man3lgrp
144 dir path=usr/share/man/man3lib
145 dir path=usr/share/man/man3mail
146 dir path=usr/share/man/man3malloc
147 dir path=usr/share/man/man3mp
148 dir path=usr/share/man/man3nsl
149 dir path=usr/share/man/man3nvpair
150 dir path=usr/share/man/man3pam
151 dir path=usr/share/man/man3pool
152 dir path=usr/share/man/man3scf
153 dir path=usr/share/man/man3sec
154 dir path=usr/share/man/man3secdb
155 dir path=usr/share/man/man3sip
156 dir path=usr/share/man/man3socket
157 dir path=usr/share/man/man3tsol
158 dir path=usr/share/man/man3uuid
159 dir path=usr/share/man/man3volmgt
160 dir path=usr/share/man/man3xcurses
161 dir path=usr/share/man/man3xnet
162 dir path=usr/share/man/man5
163 dir path=usr/share/man/man7p
164 dir path=usr/xpg4
165 dir path=usr/xpg4/lib
166 dir path=usr/xpg4/lib/${ARCH64}
167 ${i386_ONLY}file path=etc/flash/precreation/caplib group=sys mode=0500
168 file path=lib/${ARCH64}/c_synonyms.so.1
169 file path=lib/${ARCH64}/ld.so.1
170 file path=lib/${ARCH64}/libadm.so.1
171 file path=lib/${ARCH64}/libaio.so.1
172 file path=lib/${ARCH64}/libavl.so.1
173 file path=lib/${ARCH64}/libbsm.so.1
174 file path=lib/${ARCH64}/libc.so.1
175 file path=lib/${ARCH64}/libc_db.so.1
176 file path=lib/${ARCH64}/libcmdutils.so.1
177 file path=lib/${ARCH64}/libcontract.so.1
178 file path=lib/${ARCH64}/libcryptoutil.so.1
179 file path=lib/${ARCH64}/libctf.so.1
180 file path=lib/${ARCH64}/libcurses.so.1
181 file path=lib/${ARCH64}/libdevice.so.1
182 file path=lib/${ARCH64}/libdevvid.so.1
183 file path=lib/${ARCH64}/libdevinfo.so.1
184 file path=lib/${ARCH64}/libdhcputil.so.1
185 file path=lib/${ARCH64}/libdl.so.1
186 file path=lib/${ARCH64}/libdladm.so.1
187 file path=lib/${ARCH64}/libdlpi.so.1
188 file path=lib/${ARCH64}/libdoor.so.1
189 file path=lib/${ARCH64}/libefi.so.1
190 file path=lib/${ARCH64}/libelf.so.1
191 ${i386_ONLY}file path=lib/${ARCH64}/libfdisk.so.1
192 file path=lib/${ARCH64}/libgen.so.1

```

```

193 file path=lib/${ARCH64}/libinetutil.so.1
194 file path=lib/${ARCH64}/libintl.so.1
195 file path=lib/${ARCH64}/libkrmf.so.1
196 file path=lib/${ARCH64}/libkrmfberder.so.1
197 file path=lib/${ARCH64}/libkstat.so.1
198 file path=lib/${ARCH64}/libld.so.4
199 file path=lib/${ARCH64}/liblddb.so.4
200 file path=lib/${ARCH64}/libmd.so.1
201 file path=lib/${ARCH64}/libmd5.so.1
202 file path=lib/${ARCH64}/libmp.so.2
203 file path=lib/${ARCH64}/libnsl.so.1
204 file path=lib/${ARCH64}/libnvpair.so.1
205 file path=lib/${ARCH64}/libpam.so.1
206 file path=lib/${ARCH64}/libproc.so.1
207 file path=lib/${ARCH64}/libpthread.so.1
208 file path=lib/${ARCH64}/librcm.so.1
209 file path=lib/${ARCH64}/libresolv.so.2
210 file path=lib/${ARCH64}/librestart.so.1
211 file path=lib/${ARCH64}/librpcsvc.so.1
212 file path=lib/${ARCH64}/librt.so.1
213 file path=lib/${ARCH64}/librtld.so.1
214 file path=lib/${ARCH64}/librtld_db.so.1
215 file path=lib/${ARCH64}/libscf.so.1
216 file path=lib/${ARCH64}/libsec.so.1
217 file path=lib/${ARCH64}/libsecdb.so.1
218 file path=lib/${ARCH64}/libsendfile.so.1
219 file path=lib/${ARCH64}/libsocket.so.1
220 file path=lib/${ARCH64}/libsysevent.so.1
221 file path=lib/${ARCH64}/libtermcap.so.1
222 file path=lib/${ARCH64}/libthread.so.1
223 file path=lib/${ARCH64}/libtsnet.so.1
224 file path=lib/${ARCH64}/libtsol.so.2
225 file path=lib/${ARCH64}/libumem.so.1
226 file path=lib/${ARCH64}/libuuid.so.1
227 file path=lib/${ARCH64}/libutil.so.1
228 file path=lib/${ARCH64}/libw.so.1
229 file path=lib/${ARCH64}/libxnet.so.1
230 file path=lib/${ARCH64}/nss_compat.so.1
231 file path=lib/${ARCH64}/nss_dns.so.1
232 file path=lib/${ARCH64}/nss_files.so.1
233 file path=lib/${ARCH64}/nss_nis.so.1
234 file path=lib/${ARCH64}/nss_user.so.1
235 file path=lib/c_synonyms.so.1
236 file path=lib/crypto/${ARCH64}/kmf_mapper_cn.so.1
237 file path=lib/crypto/${ARCH64}/kmf_nss.so.1
238 file path=lib/crypto/${ARCH64}/kmf_openssl.so.1
239 file path=lib/crypto/${ARCH64}/kmf_pkcs11.so.1
240 file path=lib/crypto/kmf_mapper_cn.so.1
241 file path=lib/crypto/kmf_nss.so.1
242 file path=lib/crypto/kmf_openssl.so.1
243 file path=lib/crypto/kmf_pkcs11.so.1
244 file path=lib/ld.so.1
245 file path=lib/libadm.so.1
246 file path=lib/libaio.so.1
247 file path=lib/libavl.so.1
248 file path=lib/libbsm.so.1
249 file path=lib/libc.so.1 reboot-needed=true
250 file path=lib/libc_db.so.1
251 file path=lib/libcmdutils.so.1
252 file path=lib/libcontract.so.1
253 file path=lib/libcryptoutil.so.1
254 file path=lib/libctf.so.1
255 file path=lib/libcurses.so.1
256 file path=lib/libdevice.so.1
257 file path=lib/libdevvid.so.1
258 file path=lib/libdevinfo.so.1

```

```
259 file path=lib/libdhcpageant.so.1
260 file path=lib/libdhcputil.so.1
261 file path=lib/libddl.so.1
262 file path=lib/libddladm.so.1
263 file path=lib/libdldpi.so.1
264 file path=lib/libdoor.so.1
265 file path=lib/libefi.so.1
266 file path=lib/libelf.so.1
267 file path=lib/libelfsign.so.1
268 $(i386_ONLY)file path=lib/libfdisk.so.1
269 file path=lib/libgen.so.1
270 file path=lib/libinetutil.so.1
271 file path=lib/libintl.so.1
272 file path=lib/libipadm.so.1
273 file path=lib/libipmp.so.1
274 file path=lib/libkccfd.so.1
275 file path=lib/libkmf.so.1
276 file path=lib/libkmfberder.so.1
277 file path=lib/libkstat.so.1
278 file path=lib/libld.so.4
279 file path=lib/liblddbg.so.4
280 file path=lib/libmd.so.1
281 file path=lib/libmd5.so.1
282 file path=lib/libmp.so.1
283 file path=lib/libmp.so.2
284 file path=lib/libnsl.so.1
285 file path=lib/libnvpair.so.1
286 file path=lib/libnwm.so.1
287 file path=lib/libpam.so.1
288 file path=lib/libproc.so.1
289 file path=lib/libpthread.so.1
290 file path=lib/librcm.so.1
291 file path=lib/libresolv.so.1
292 file path=lib/libresolv.so.2
293 file path=lib/librestart.so.1
294 file path=lib/librpcsvc.so.1
295 file path=lib/librt.so.1
296 file path=lib/librtdb.so.1
297 file path=lib/librtdb_db.so.1
298 file path=lib/libscf.so.1
299 file path=lib/libsec.so.1
300 file path=lib/libsecdb.so.1
301 file path=lib/libsendfile.so.1
302 file path=lib/libsocket.so.1
303 file path=lib/libsysevent.so.1
304 file path=lib/libtermcap.so.1
305 file path=lib/libthread.so.1
306 file path=lib/libtsnet.so.1
307 file path=lib/libtsol.so.2
308 file path=lib/libumem.so.1
309 file path=lib/libuuid.so.1
310 file path=lib/libuutil.so.1
311 file path=lib/libw.so.1
312 file path=lib/libxnet.so.1
313 file path=lib/mpxio/stmsboot_util mode=0555
314 file path=lib/nss_compat.so.1
315 file path=lib/nss_dns.so.1
316 file path=lib/nss_files.so.1
317 file path=lib/nss_nis.so.1
318 file path=lib/nss_user.so.1
319 file path=usr/lib/$(ARCH64)/0@0.so.1
320 file path=usr/lib/$(ARCH64)/getloginx.so.1
321 file path=usr/lib/$(ARCH64)/libadutils.so.1
322 file path=usr/lib/$(ARCH64)/libast.so.1
323 file path=usr/lib/$(ARCH64)/libbsdmalloc.so.1
324 file path=usr/lib/$(ARCH64)/libcfgadm.so.1
```

```
325 file path=usr/lib/$(ARCH64)/libcmd.so.1
326 file path=usr/lib/$(ARCH64)/libcommutil.so.1
327 file path=usr/lib/$(ARCH64)/libcrle.so.1
328 file path=usr/lib/$(ARCH64)/libcrypt.so.1
329 file path=usr/lib/$(ARCH64)/libdisasm.so.1
330 file path=usr/lib/$(ARCH64)/libdll.so.1
331 file path=usr/lib/$(ARCH64)/libexacct.so.1
332 file path=usr/lib/$(ARCH64)/libform.so.1
333 file path=usr/lib/$(ARCH64)/libfstyp.so.1
334 file path=usr/lib/$(ARCH64)/libhotplug.so.1
335 file path=usr/lib/$(ARCH64)/libidmap.so.1
336 file path=usr/lib/$(ARCH64)/libike.so.1
337 file path=usr/lib/$(ARCH64)/libipmi.so.1
338 file path=usr/lib/$(ARCH64)/libipp.so.1
339 file path=usr/lib/$(ARCH64)/libipsecutil.so.1
340 file path=usr/lib/$(ARCH64)/libkvm.so.1
341 file path=usr/lib/$(ARCH64)/libl.so.1
342 file path=usr/lib/$(ARCH64)/libldap.so.5
343 file path=usr/lib/$(ARCH64)/liblgrp.so.1
344 file path=usr/lib/$(ARCH64)/liblm.so.1
345 file path=usr/lib/$(ARCH64)/libmail.so.1
346 file path=usr/lib/$(ARCH64)/libmalloc.so.1
347 file path=usr/lib/$(ARCH64)/libmapmalloc.so.1
348 file path=usr/lib/$(ARCH64)/libmenu.so.1
349 file path=usr/lib/$(ARCH64)/libmtmalloc.so.1
350 file path=usr/lib/$(ARCH64)/libnls.so.1
351 file path=usr/lib/$(ARCH64)/libpanel.so.1
352 file path=usr/lib/$(ARCH64)/libpkcs11.so.1
353 file path=usr/lib/$(ARCH64)/libproject.so.1
354 file path=usr/lib/$(ARCH64)/libraidcfg.so.1
355 file path=usr/lib/$(ARCH64)/librepase.so.1
356 file path=usr/lib/$(ARCH64)/libsched.so.1
357 file path=usr/lib/$(ARCH64)/libsctp.so.1
358 file path=usr/lib/$(ARCH64)/libshell.so.1
359 file path=usr/lib/$(ARCH64)/libsip.so.1
360 file path=usr/lib/$(ARCH64)/libslldap.so.1
361 file path=usr/lib/$(ARCH64)/libsembios.so.1
362 file path=usr/lib/$(ARCH64)/libsoftcrypto.so.1
363 file path=usr/lib/$(ARCH64)/libsum.so.1
364 $(sparc_ONLY)file path=usr/lib/$(ARCH64)/libv12n.so.1
365 file path=usr/lib/$(ARCH64)/libvolmgt.so.1
366 file path=usr/lib/$(ARCH64)/liby.so.1
367 file path=usr/lib/$(ARCH64)/libzoneinfo.so.1
368 file path=usr/lib/$(ARCH64)/nss_ad.so.1
369 file path=usr/lib/$(ARCH64)/nss_ldap.so.1
370 file path=usr/lib/$(ARCH64)/passwdutil.so.1
371 file path=usr/lib/$(ARCH64)/straddr.so.2
372 file path=usr/lib/$(ARCH64)/watchmalloc.so.1
373 file path=usr/lib/0@0.so.1
374 file path=usr/lib/cfgadm/$(ARCH64)/ib.so.1
375 file path=usr/lib/cfgadm/$(ARCH64)/pci.so.1
376 $(i386_ONLY)file path=usr/lib/cfgadm/$(ARCH64)/sata.so.1
377 file path=usr/lib/cfgadm/$(ARCH64)/scsi.so.1
378 file path=usr/lib/cfgadm/$(ARCH64)/shp.so.1
379 file path=usr/lib/cfgadm/$(ARCH64)/usb.so.1
380 file path=usr/lib/cfgadm/ib.so.1
381 file path=usr/lib/cfgadm/pci.so.1
382 $(i386_ONLY)file path=usr/lib/cfgadm/sata.so.1
383 file path=usr/lib/cfgadm/scsi.so.1
384 file path=usr/lib/cfgadm/shp.so.1
385 file path=usr/lib/cfgadm/usb.so.1
386 file path=usr/lib/extendedFILE.so.1
387 file path=usr/lib/getloginx.so.1
388 file path=usr/lib/lib.b mode=0444
389 file path=usr/lib/libadutils.so.1
390 file path=usr/lib/libast.so.1
```

```

391 file path=usr/lib/libbsdmalloc.so.1
392 $(i386_ONLY)file path=usr/lib/libc/libc_hwcap1.so.1 reboot-needed=true
393 $(i386_ONLY)file path=usr/lib/libc/libc_hwcap2.so.1 reboot-needed=true
394 $(i386_ONLY)file path=usr/lib/libc/libc_hwcap3.so.1 reboot-needed=true
395 file path=usr/lib/libcfdm.so.1
396 file path=usr/lib/libcmd.so.1
397 file path=usr/lib/libcommutil.so.1
398 file path=usr/lib/libcrl.so.1
399 file path=usr/lib/libcrypt.so.1
400 file path=usr/lib/libdisasm.so.1
401 file path=usr/lib/libdll.so.1
402 file path=usr/lib/libexacct.so.1
403 file path=usr/lib/libform.so.1
404 file path=usr/lib/libfstyp.so.1
405 file path=usr/lib/libhotplug.so.1
406 file path=usr/lib/libidmap.so.1
407 file path=usr/lib/libike.so.1
408 file path=usr/lib/libinetsvc.so.1
409 file path=usr/lib/libipmi.so.1
410 file path=usr/lib/libipp.so.1
411 file path=usr/lib/libipsecutil.so.1
412 file path=usr/lib/libkvm.so.1
413 file path=usr/lib/libl.so.1
414 file path=usr/lib/libldap.so.5
415 file path=usr/lib/liblgrp.so.1
416 file path=usr/lib/liblm.so.1
417 file path=usr/lib/libmail.so.1
418 file path=usr/lib/libmalloc.so.1
419 file path=usr/lib/libmapmalloc.so.1
420 file path=usr/lib/libmenu.so.1
421 file path=usr/lib/libmtmalloc.so.1
422 file path=usr/lib/libnls.so.1
423 file path=usr/lib/libpanel.so.1
424 file path=usr/lib/libpkcs11.so.1
425 file path=usr/lib/libproject.so.1
426 file path=usr/lib/libraidcfg.so.1
427 file path=usr/lib/librepase.so.1
428 file path=usr/lib/libsched.so.1
429 file path=usr/lib/libsectp.so.1
430 file path=usr/lib/libshell.so.1
431 file path=usr/lib/libsip.so.1
432 file path=usr/lib/libslldap.so.1
433 file path=usr/lib/libsmbios.so.1
434 file path=usr/lib/libsoftcrypto.so.1
435 file path=usr/lib/libsum.so.1
436 file path=usr/lib/libsys.so.1
437 $(sparc_ONLY)file path=usr/lib/libv12n.so.1
438 file path=usr/lib/libvolmgt.so.1
439 file path=usr/lib/libwrap.so.1.0
440 file path=usr/lib/liby.so.1
441 file path=usr/lib/libzoneinfo.so.1
442 file path=usr/lib/nss_ad.so.1
443 file path=usr/lib/nss_ldap.so.1
444 file path=usr/lib/passwdutil.so.1
445 file path=usr/lib/python2.6/vendor-packages/solaris/__init__.py
446 file path=usr/lib/python2.6/vendor-packages/solaris/__init__.pyc
447 file path=usr/lib/python2.6/vendor-packages/solaris/misc.so
448 file path=usr/lib/raidcfg/(ARCH64)/mpt.so.1
449 file path=usr/lib/raidcfg/mpt.so.1
450 file path=usr/lib/scsi/(ARCH64)/libscsi.so.1
451 file path=usr/lib/scsi/(ARCH64)/libsos.so.1
452 file path=usr/lib/scsi/(ARCH64)/libsmp.so.1
453 file path=usr/lib/scsi/libscsi.so.1
454 file path=usr/lib/scsi/libsos.so.1
455 file path=usr/lib/scsi/libsmp.so.1
456 file path=usr/lib/scsi/plugins/scsi/engines/(ARCH64)/uscsi.so

```

```

457 file path=usr/lib/scsi/plugins/scsi/engines/uscsi.so
458 file path=usr/lib/scsi/plugins/ses/framework/(ARCH64)/libsos.so
459 file path=usr/lib/scsi/plugins/ses/framework/(ARCH64)/ses2.so
460 file path=usr/lib/scsi/plugins/ses/framework/libses.so
461 file path=usr/lib/scsi/plugins/ses/framework/ses2.so
462 file path=usr/lib/scsi/plugins/smp/engine/(ARCH64)/usmp.so
463 file path=usr/lib/scsi/plugins/smp/engine/usmp.so
464 file path=usr/lib/scsi/plugins/smp/framework/(ARCH64)/sas2.so
465 file path=usr/lib/scsi/plugins/smp/framework/sas2.so
466 file path=usr/lib/security/(ARCH64)/crypt_bsdbf.so.1
467 file path=usr/lib/security/(ARCH64)/crypt_bsdmd5.so.1
468 file path=usr/lib/security/(ARCH64)/crypt_sha256.so.1
469 file path=usr/lib/security/(ARCH64)/crypt_sha512.so.1
470 file path=usr/lib/security/(ARCH64)/crypt_sunmd5.so.1
471 file path=usr/lib/security/(ARCH64)/pam_allow.so.1
472 file path=usr/lib/security/(ARCH64)/pam_authtok_check.so.1
473 file path=usr/lib/security/(ARCH64)/pam_authtok_get.so.1
474 file path=usr/lib/security/(ARCH64)/pam_authtok_store.so.1
475 file path=usr/lib/security/(ARCH64)/pam_deny.so.1
476 file path=usr/lib/security/(ARCH64)/pam_dhkeys.so.1
477 file path=usr/lib/security/(ARCH64)/pam_dial_auth.so.1
478 file path=usr/lib/security/(ARCH64)/pam_ldap.so.1
479 file path=usr/lib/security/(ARCH64)/pam_list.so.1
480 file path=usr/lib/security/(ARCH64)/pam_passwd_auth.so.1
481 file path=usr/lib/security/(ARCH64)/pam_rhosts_auth.so.1
482 file path=usr/lib/security/(ARCH64)/pam_roles.so.1
483 file path=usr/lib/security/(ARCH64)/pam_sample.so.1
484 file path=usr/lib/security/(ARCH64)/pam_tsol_account.so.1
485 file path=usr/lib/security/(ARCH64)/pam_unix_account.so.1
486 file path=usr/lib/security/(ARCH64)/pam_unix_auth.so.1
487 file path=usr/lib/security/(ARCH64)/pam_unix_cred.so.1
488 file path=usr/lib/security/(ARCH64)/pam_unix_session.so.1
489 file path=usr/lib/security/(ARCH64)/pkcs11_kernel.so.1
490 file path=usr/lib/security/(ARCH64)/pkcs11_softtoken.so.1
491 file path=usr/lib/security/(ARCH64)/pkcs11_tpm.so.1
492 file path=usr/lib/security/audit_binfile.so.1
493 file path=usr/lib/security/audit_remote.so.1
494 file path=usr/lib/security/audit_syslog.so.1
495 file path=usr/lib/security/crypt_bsdbf.so.1
496 file path=usr/lib/security/crypt_bsdmd5.so.1
497 file path=usr/lib/security/crypt_sha256.so.1
498 file path=usr/lib/security/crypt_sha512.so.1
499 file path=usr/lib/security/crypt_sunmd5.so.1
500 file path=usr/lib/security/pam_allow.so.1
501 file path=usr/lib/security/pam_authtok_check.so.1
502 file path=usr/lib/security/pam_authtok_get.so.1
503 file path=usr/lib/security/pam_authtok_store.so.1
504 file path=usr/lib/security/pam_deny.so.1
505 file path=usr/lib/security/pam_dhkeys.so.1
506 file path=usr/lib/security/pam_dial_auth.so.1
507 file path=usr/lib/security/pam_ldap.so.1
508 file path=usr/lib/security/pam_list.so.1
509 file path=usr/lib/security/pam_passwd_auth.so.1
510 file path=usr/lib/security/pam_rhosts_auth.so.1
511 file path=usr/lib/security/pam_roles.so.1
512 file path=usr/lib/security/pam_sample.so.1
513 file path=usr/lib/security/pam_tsol_account.so.1
514 file path=usr/lib/security/pam_unix_account.so.1
515 file path=usr/lib/security/pam_unix_auth.so.1
516 file path=usr/lib/security/pam_unix_cred.so.1
517 file path=usr/lib/security/pam_unix_session.so.1
518 file path=usr/lib/security/pkcs11_kernel.so.1
519 file path=usr/lib/security/pkcs11_softtoken.so.1
520 file path=usr/lib/security/pkcs11_tpm.so.1
521 file path=usr/lib/straddr.so.2
522 file path=usr/lib/watchmalloc.so.1

```

```

523 # XXX: Obsoleted by open il8n?
524 file path=usr/xpg4/lib/$(ARCH64)/libcurses.so.1
525 file path=usr/xpg4/lib/$(ARCH64)/libcurses.so.2
526 file path=usr/xpg4/lib/libcurses.so.1
527 file path=usr/xpg4/lib/libcurses.so.2
528 legacy pkg=SUNWcsl \
529     desc="core shared libraries for a specific instruction-set architecture" \
530     name="Core Solaris, (Shared Libs)"
531 legacy pkg=SUNWcslr \
532     desc="core software for a specific instruction-set architecture" \
533     name="Core Solaris Libraries (Root)"
534 license cr_Sun license=cr_Sun
535 license lic_CDDL license=lic_CDDL
536 license lic_OSBL license=lic_OSBL
537 license lic_OSBL_preamble license=lic_OSBL_preamble
538 # libwrap is part of tcp wrappers along with tcpd
539 license usr/src/cmd/tcpd/THIRDPARTYLICENSE \
540     license=usr/src/cmd/tcpd/THIRDPARTYLICENSE
541 license usr/src/common/crypto/THIRDPARTYLICENSE.cryptogams \
542     license=usr/src/common/crypto/THIRDPARTYLICENSE.cryptogams
543 license usr/src/common/crypto/aes/amd64/THIRDPARTYLICENSE.gladman \
544     license=usr/src/common/crypto/aes/amd64/THIRDPARTYLICENSE.gladman
545 license usr/src/common/crypto/aes/amd64/THIRDPARTYLICENSE.openssl \
546     license=usr/src/common/crypto/aes/amd64/THIRDPARTYLICENSE.openssl
547 license usr/src/common/crypto/ecc/THIRDPARTYLICENSE \
548     license=usr/src/common/crypto/ecc/THIRDPARTYLICENSE
549 license usr/src/common/crypto/md5/amd64/THIRDPARTYLICENSE \
550     license=usr/src/common/crypto/md5/amd64/THIRDPARTYLICENSE
551 license usr/src/common/mpi/THIRDPARTYLICENSE \
552     license=usr/src/common/mpi/THIRDPARTYLICENSE
553 license usr/src/lib/libast/THIRDPARTYLICENSE \
554     license=usr/src/lib/libast/THIRDPARTYLICENSE
555 license usr/src/lib/libbsdmalloc/THIRDPARTYLICENSE \
556     license=usr/src/lib/libbsdmalloc/THIRDPARTYLICENSE
557 license usr/src/lib/libc/THIRDPARTYLICENSE \
558     license=usr/src/lib/libc/THIRDPARTYLICENSE
559 license usr/src/lib/libcmd/THIRDPARTYLICENSE \
560     license=usr/src/lib/libcmd/THIRDPARTYLICENSE
561 license usr/src/lib/libdll/THIRDPARTYLICENSE \
562     license=usr/src/lib/libdll/THIRDPARTYLICENSE
563 license usr/src/lib/libinetutil/common/THIRDPARTYLICENSE \
564     license=usr/src/lib/libinetutil/common/THIRDPARTYLICENSE
565 license usr/src/lib/libkmf/THIRDPARTYLICENSE \
566     license=usr/src/lib/libkmf/THIRDPARTYLICENSE
567 license usr/src/lib/libldap5/THIRDPARTYLICENSE \
568     license=usr/src/lib/libldap5/THIRDPARTYLICENSE
569 license usr/src/lib/libmp/common/THIRDPARTYLICENSE \
570     license=usr/src/lib/libmp/common/THIRDPARTYLICENSE
571 license usr/src/lib/libresolv/THIRDPARTYLICENSE \
572     license=usr/src/lib/libresolv/THIRDPARTYLICENSE
573 license usr/src/lib/libresolv2/THIRDPARTYLICENSE \
574     license=usr/src/lib/libresolv2/THIRDPARTYLICENSE
575 license usr/src/lib/libshell/THIRDPARTYLICENSE \
576     license=usr/src/lib/libshell/THIRDPARTYLICENSE
577 license usr/src/lib/libsum/THIRDPARTYLICENSE \
578     license=usr/src/lib/libsum/THIRDPARTYLICENSE
579 license usr/src/lib/pam_modules/authok_check/THIRDPARTYLICENSE \
580     license=usr/src/lib/pam_modules/authok_check/THIRDPARTYLICENSE
581 license usr/src/lib/passwdutil/THIRDPARTYLICENSE \
582     license=usr/src/lib/passwdutil/THIRDPARTYLICENSE
583 license usr/src/lib/pkcs11/pkcs11_tpm/THIRDPARTYLICENSE \
584     license=usr/src/lib/pkcs11/pkcs11_tpm/THIRDPARTYLICENSE
585 license usr/src/uts/common/sys/THIRDPARTYLICENSE.unicode \
586     license=usr/src/uts/common/sys/THIRDPARTYLICENSE.unicode
587 link path=lib/$(ARCH64)/libadm.so target=libadm.so.1
588 link path=lib/$(ARCH64)/libaio.so target=libaio.so.1

```

```

589 link path=lib/$(ARCH64)/libbsm.so target=libbsm.so.1
590 link path=lib/$(ARCH64)/libc.so reboot-needed=true target=libc.so.1
591 link path=lib/$(ARCH64)/libc_db.so target=libc_db.so.1
592 link path=lib/$(ARCH64)/libcontract.so target=libcontract.so.1
593 link path=lib/$(ARCH64)/libcryptoutil.so target=libcryptoutil.so.1
594 link path=lib/$(ARCH64)/libctf.so target=libctf.so.1
595 link path=lib/$(ARCH64)/libcurses.so target=libcurses.so.1
596 link path=lib/$(ARCH64)/libdevice.so target=libdevice.so.1
597 link path=lib/$(ARCH64)/libdevinfo.so target=libdevinfo.so.1
598 link path=lib/$(ARCH64)/libdevinfo.so target=libdevinfo.so.1
599 link path=lib/$(ARCH64)/libdl.so target=libdl.so.1
600 link path=lib/$(ARCH64)/libdladm.so target=libdladm.so.1
601 link path=lib/$(ARCH64)/libdlpi.so target=libdlpi.so.1
602 link path=lib/$(ARCH64)/libdoor.so target=libdoor.so.1
603 link path=lib/$(ARCH64)/libefi.so target=libefi.so.1
604 link path=lib/$(ARCH64)/libelf.so target=libelf.so.1
605 $(i386_ONLY)link path=lib/$(ARCH64)/libfdisk.so target=libfdisk.so.1
606 link path=lib/$(ARCH64)/libgen.so target=libgen.so.1
607 link path=lib/$(ARCH64)/libintl.so target=libintl.so.1
608 link path=lib/$(ARCH64)/libkmf.so target=libkmf.so.1
609 link path=lib/$(ARCH64)/libkmfberder.so target=libkmfberder.so.1
610 link path=lib/$(ARCH64)/libkstat.so target=libkstat.so.1
611 link path=lib/$(ARCH64)/libmd.so target=libmd.so.1
612 link path=lib/$(ARCH64)/libmd5.so target=libmd5.so.1
613 link path=lib/$(ARCH64)/libmp.so target=libmp.so.2
614 link path=lib/$(ARCH64)/libnsl.so target=libnsl.so.1
615 link path=lib/$(ARCH64)/libnvpair.so target=libnvpair.so.1
616 link path=lib/$(ARCH64)/libpam.so target=libpam.so.1
617 link path=lib/$(ARCH64)/libposix4.so target=libposix4.so.1
618 link path=lib/$(ARCH64)/libposix4.so.1 target=librt.so.1
619 link path=lib/$(ARCH64)/libproc.so target=libproc.so.1
620 link path=lib/$(ARCH64)/libpthread.so target=libpthread.so.1
621 link path=lib/$(ARCH64)/librcm.so target=librcm.so.1
622 link path=lib/$(ARCH64)/libresolv.so target=libresolv.so.2
623 link path=lib/$(ARCH64)/librestart.so target=librestart.so.1
624 link path=lib/$(ARCH64)/librpcsvc.so target=librpcsvc.so.1
625 link path=lib/$(ARCH64)/librt.so target=librt.so.1
626 link path=lib/$(ARCH64)/librtld_db.so target=librtld_db.so.1
627 link path=lib/$(ARCH64)/libscaf.so target=libscaf.so.1
628 link path=lib/$(ARCH64)/libsec.so target=libsec.so.1
629 link path=lib/$(ARCH64)/libsecdb.so target=libsecdb.so.1
630 link path=lib/$(ARCH64)/libsendfile.so target=libsendfile.so.1
631 link path=lib/$(ARCH64)/libsocket.so target=libsocket.so.1
632 link path=lib/$(ARCH64)/libsysevent.so target=libsysevent.so.1
633 link path=lib/$(ARCH64)/libtermcap.so target=libtermcap.so.1
634 link path=lib/$(ARCH64)/libtermplib.so target=libtermplib.so.1
635 link path=lib/$(ARCH64)/libtermplib.so.1 target=libcurses.so.1
636 link path=lib/$(ARCH64)/libthread.so target=libthread.so.1
637 link path=lib/$(ARCH64)/libthread_db.so target=lib_db.so.1
638 link path=lib/$(ARCH64)/libthread_db.so.1 target=libc_db.so.1
639 link path=lib/$(ARCH64)/libtsnet.so target=libtsnet.so.1
640 link path=lib/$(ARCH64)/libtsol.so target=libtsol.so.2
641 link path=lib/$(ARCH64)/libumem.so target=libumem.so.1
642 link path=lib/$(ARCH64)/libuuid.so target=libuuid.so.1
643 link path=lib/$(ARCH64)/libuutil.so target=libuutil.so.1
644 link path=lib/$(ARCH64)/libw.so target=libw.so.1
645 link path=lib/$(ARCH64)/libxnet.so target=libxnet.so.1
646 link path=lib/32 target=.
647 link path=lib/64 target=$(ARCH64)
648 link path=lib/crypto/32 target=.
649 link path=lib/crypto/64 target=$(ARCH64)
650 link path=lib/libadm.so target=libadm.so.1
651 link path=lib/libaio.so target=libaio.so.1
652 link path=lib/libbsm.so target=libbsm.so.1
653 link path=lib/libc.so target=libc.so.1
654 link path=lib/libc_db.so target=libc_db.so.1

```



```

655 link path=lib/libcontract.so target=libcontract.so.1
656 link path=lib/libcryptoutil.so target=libcryptoutil.so.1
657 link path=lib/libctf.so target=libctf.so.1
658 link path=lib/libcurses.so target=libcurses.so.1
659 link path=lib/libdevice.so target=libdevice.so.1
660 link path=lib/libdevid.so target=libdevid.so.1
661 link path=lib/libdevinfo.so target=libdevinfo.so.1
662 link path=lib/libdl.so target=libdl.so.1
663 link path=lib/libdladm.so target=libdladm.so.1
664 link path=lib/libdmpi.so target=libdmpi.so.1
665 link path=lib/libdoor.so target=libdoor.so.1
666 link path=lib/libefi.so target=libefi.so.1
667 link path=lib/libelf.so target=libelf.so.1
668 link path=lib/libelfsign.so target=libelfsign.so.1
669 link path=lib/libfdisk.so target=libfdisk.so.1
670 link path=lib/libgen.so target=libgen.so.1
671 link path=lib/libintl.so target=libintl.so.1
672 link path=lib/libipmp.so target=libipmp.so.1
673 link path=lib/libkrmf.so target=libkrmf.so.1
674 link path=lib/libkrmfborder.so target=libkrmfborder.so.1
675 link path=lib/libkstat.so target=libkstat.so.1
676 link path=lib/libmd.so target=libmd.so.1
677 link path=lib/libmd5.so target=libmd5.so.1
678 link path=lib/libmp.so target=libmp.so.2
679 link path=lib/libnsl.so target=libnsl.so.1
680 link path=lib/libnvpair.so target=libnvpair.so.1
681 link path=lib/libnwam.so target=libnwam.so.1
682 link path=lib/libpam.so target=libpam.so.1
683 link path=lib/libposix4.so target=libposix4.so.1
684 link path=lib/libposix4.so.1 target=librt.so.1
685 link path=lib/libproc.so target=libproc.so.1
686 link path=lib/libpthread.so target=libpthread.so.1
687 link path=lib/librcm.so target=librcm.so.1
688 link path=lib/libresolv.so target=libresolv.so.2
689 link path=lib/librpcsvc.so target=librpcsvc.so.1
690 link path=lib/librt.so target=librt.so.1
691 link path=lib/librtld_db.so target=librtld_db.so.1
692 link path=lib/libscf.so target=libscf.so.1
693 link path=lib/libsec.so target=libsec.so.1
694 link path=lib/libsecdb.so target=libsecdb.so.1
695 link path=lib/libsendfile.so target=libsendfile.so.1
696 link path=lib/libsocket.so target=libsocket.so.1
697 link path=lib/libsysevent.so target=libsysevent.so.1
698 link path=lib/libtermcap.so target=libtermcap.so.1
699 link path=lib/libtermplib.so target=libtermplib.so.1
700 link path=lib/libtermplib.so.1 target=libcurses.so.1
701 link path=lib/libthread.so target=libthread.so.1
702 link path=lib/libthread_db.so target=libthread_db.so.1
703 link path=lib/libthread_db.so.1 target=libc_db.so.1
704 link path=lib/libtsol.so target=libtsol.so.2
705 link path=lib/libumem.so target=libumem.so.1
706 link path=lib/libuuid.so target=libuuid.so.1
707 link path=lib/libw.so target=libw.so.1
708 link path=lib/libxnet.so target=libxnet.so.1
709 link path=lib/libsecure/32 target=
710 link path=lib/libsecure/64 target=$(ARCH64)
711 link path=usr/ccs/lib/$(ARCH64)/libcurses.so \
712 target=usr/lib/$(ARCH64)/libcurses.so.1
713 link path=usr/ccs/lib/$(ARCH64)/libform.so \
714 target=usr/lib/$(ARCH64)/libform.so.1
715 link path=usr/ccs/lib/$(ARCH64)/libgen.so \
716 target=usr/lib/$(ARCH64)/libgen.so.1
717 link path=usr/ccs/lib/$(ARCH64)/libl.so \
718 target=usr/lib/$(ARCH64)/libl.so.1
719 link path=usr/ccs/lib/$(ARCH64)/libmalloc.so \
720 target=usr/lib/$(ARCH64)/libmalloc.so.1

```

```

721 link path=usr/ccs/lib/$(ARCH64)/libmenu.so \
722 target=usr/lib/$(ARCH64)/libmenu.so.1
723 link path=usr/ccs/lib/$(ARCH64)/libpanel.so \
724 target=usr/lib/$(ARCH64)/libpanel.so.1
725 link path=usr/ccs/lib/$(ARCH64)/libtermcap.so \
726 target=usr/lib/$(ARCH64)/libtermcap.so.1
727 link path=usr/ccs/lib/$(ARCH64)/libtermplib.so \
728 target=usr/lib/$(ARCH64)/libtermplib.so.1
729 link path=usr/ccs/lib/$(ARCH64)/liby.so \
730 target=usr/lib/$(ARCH64)/liby.so.1
731 link path=usr/ccs/lib/libcurses.so target=usr/lib/libcurses.so.1
732 link path=usr/ccs/lib/libform.so target=usr/lib/libform.so.1
733 link path=usr/ccs/lib/libgen.so target=usr/lib/libgen.so.1
734 link path=usr/ccs/lib/libl.so target=usr/lib/libl.so.1
735 link path=usr/ccs/lib/libmalloc.so target=usr/lib/libmalloc.so.1
736 link path=usr/ccs/lib/libmenu.so target=usr/lib/libmenu.so.1
737 link path=usr/ccs/lib/libpanel.so target=usr/lib/libpanel.so.1
738 link path=usr/ccs/lib/libtermcap.so target=usr/lib/libtermcap.so.1
739 link path=usr/ccs/lib/libtermplib.so target=usr/lib/libtermplib.so.1
740 link path=usr/ccs/lib/liby.so target=usr/lib/liby.so.1
741 link path=usr/lib/$(ARCH64)/libadm.so \
742 target=usr/lib/$(ARCH64)/libadm.so.1
743 link path=usr/lib/$(ARCH64)/libadm.so.1 \
744 target=usr/lib/$(ARCH64)/libadm.so.1
745 link path=usr/lib/$(ARCH64)/libadutils.so target=libadutils.so.1
746 link path=usr/lib/$(ARCH64)/libaio.so \
747 target=usr/lib/$(ARCH64)/libaio.so.1
748 link path=usr/lib/$(ARCH64)/libaio.so.1 \
749 target=usr/lib/$(ARCH64)/libaio.so.1
750 link path=usr/lib/$(ARCH64)/libavl.so.1 \
751 target=usr/lib/$(ARCH64)/libavl.so.1
752 link path=usr/lib/$(ARCH64)/libbsdmalloc.so target=libbsdmalloc.so.1
753 link path=usr/lib/$(ARCH64)/libbsm.so \
754 target=usr/lib/$(ARCH64)/libbsm.so.1
755 link path=usr/lib/$(ARCH64)/libbsm.so.1 \
756 target=usr/lib/$(ARCH64)/libbsm.so.1
757 link path=usr/lib/$(ARCH64)/libc.so target=usr/lib/$(ARCH64)/libc.so.1
758 link path=usr/lib/$(ARCH64)/libc.so.1 target=usr/lib/$(ARCH64)/libc.so.1
759 link path=usr/lib/$(ARCH64)/libc_db.so \
760 target=usr/lib/$(ARCH64)/libc_db.so.1
761 link path=usr/lib/$(ARCH64)/libc_db.so.1 \
762 target=usr/lib/$(ARCH64)/libc_db.so.1
763 link path=usr/lib/$(ARCH64)/libcfgadm.so target=libcfgadm.so.1
764 link path=usr/lib/$(ARCH64)/libcmd.so target=libcmd.so.1
765 link path=usr/lib/$(ARCH64)/libcmdutils.so.1 \
766 target=usr/lib/$(ARCH64)/libcmdutils.so.1
767 link path=usr/lib/$(ARCH64)/libcommutil.so target=libcommutil.so.1
768 link path=usr/lib/$(ARCH64)/libcontract.so \
769 target=usr/lib/$(ARCH64)/libcontract.so.1
770 link path=usr/lib/$(ARCH64)/libcontract.so.1 \
771 target=usr/lib/$(ARCH64)/libcontract.so.1
772 link path=usr/lib/$(ARCH64)/libcrypt.so target=libcrypt.so.1
773 link path=usr/lib/$(ARCH64)/libcrypt_d.so target=libcrypt.so
774 link path=usr/lib/$(ARCH64)/libcrypt_d.so.1 target=libcrypt.so.1
775 link path=usr/lib/$(ARCH64)/libcrypt_i.so target=libcrypt.so
776 link path=usr/lib/$(ARCH64)/libcrypt_i.so.1 target=libcrypt.so.1
777 link path=usr/lib/$(ARCH64)/libctf.so \
778 target=usr/lib/$(ARCH64)/libctf.so.1
779 link path=usr/lib/$(ARCH64)/libctf.so.1 \
780 target=usr/lib/$(ARCH64)/libctf.so.1
781 link path=usr/lib/$(ARCH64)/libcurses.so \
782 target=usr/lib/$(ARCH64)/libcurses.so.1
783 link path=usr/lib/$(ARCH64)/libcurses.so.1 \
784 target=usr/lib/$(ARCH64)/libcurses.so.1
785 link path=usr/lib/$(ARCH64)/libdevice.so \
786 target=usr/lib/$(ARCH64)/libdevice.so.1

```

```

787 link path=usr/lib/${ARCH64}/libdevice.so.1 \
788     target=../../../../lib/${ARCH64}/libdevice.so.1
789 link path=usr/lib/${ARCH64}/libdevinfo.so \
790     target=../../../../lib/${ARCH64}/libdevinfo.so.1
791 link path=usr/lib/${ARCH64}/libdevinfo.so.1 \
792     target=../../../../lib/${ARCH64}/libdevinfo.so.1
793 link path=usr/lib/${ARCH64}/libdevinfo.so \
794     target=../../../../lib/${ARCH64}/libdevinfo.so.1
795 link path=usr/lib/${ARCH64}/libdevinfo.so.1 \
796     target=../../../../lib/${ARCH64}/libdevinfo.so.1
797 link path=usr/lib/${ARCH64}/libdhcputil.so.1 \
798     target=../../../../lib/${ARCH64}/libdhcputil.so.1
799 link path=usr/lib/${ARCH64}/libdisasm.so target=libdisasm.so.1
800 link path=usr/lib/${ARCH64}/libdl.so target=../../../../lib/${ARCH64}/libdl.so.1
801 link path=usr/lib/${ARCH64}/libdl.so.1 \
802     target=../../../../lib/${ARCH64}/libdl.so.1
803 link path=usr/lib/${ARCH64}/libdlpi.so \
804     target=../../../../lib/${ARCH64}/libdlpi.so.1
805 link path=usr/lib/${ARCH64}/libdlpi.so.1 \
806     target=../../../../lib/${ARCH64}/libdlpi.so.1
807 link path=usr/lib/${ARCH64}/libdoor.so \
808     target=../../../../lib/${ARCH64}/libdoor.so.1
809 link path=usr/lib/${ARCH64}/libdoor.so.1 \
810     target=../../../../lib/${ARCH64}/libdoor.so.1
811 link path=usr/lib/${ARCH64}/libefi.so \
812     target=../../../../lib/${ARCH64}/libefi.so.1
813 link path=usr/lib/${ARCH64}/libefi.so.1 \
814     target=../../../../lib/${ARCH64}/libefi.so.1
815 link path=usr/lib/${ARCH64}/libelf.so \
816     target=../../../../lib/${ARCH64}/libelf.so.1
817 link path=usr/lib/${ARCH64}/libelf.so.1 \
818     target=../../../../lib/${ARCH64}/libelf.so.1
819 link path=usr/lib/${ARCH64}/libexacct.so target=libexacct.so.1
820 ${i386_ONLY}link path=usr/lib/${ARCH64}/libfdisk.so \
821     target=../../../../lib/${ARCH64}/libfdisk.so.1
822 ${i386_ONLY}link path=usr/lib/${ARCH64}/libfdisk.so.1 \
823     target=../../../../lib/${ARCH64}/libfdisk.so.1
824 link path=usr/lib/${ARCH64}/libform.so target=libform.so.1
825 link path=usr/lib/${ARCH64}/libfstyp.so target=libfstyp.so.1
826 link path=usr/lib/${ARCH64}/libgen.so \
827     target=../../../../lib/${ARCH64}/libgen.so.1
828 link path=usr/lib/${ARCH64}/libgen.so.1 \
829     target=../../../../lib/${ARCH64}/libgen.so.1
830 link path=usr/lib/${ARCH64}/libhotplug.so target=libhotplug.so.1
831 link path=usr/lib/${ARCH64}/libidmap.so target=libidmap.so.1
832 link path=usr/lib/${ARCH64}/libinetutil.so.1 \
833     target=../../../../lib/${ARCH64}/libinetutil.so.1
834 link path=usr/lib/${ARCH64}/libintl.so \
835     target=../../../../lib/${ARCH64}/libintl.so.1
836 link path=usr/lib/${ARCH64}/libintl.so.1 \
837     target=../../../../lib/${ARCH64}/libintl.so.1
838 link path=usr/lib/${ARCH64}/libipmi.so target=libipmi.so.1
839 link path=usr/lib/${ARCH64}/libipp.so target=libipp.so.1
840 link path=usr/lib/${ARCH64}/libkstat.so \
841     target=../../../../lib/${ARCH64}/libkstat.so.1
842 link path=usr/lib/${ARCH64}/libkstat.so.1 \
843     target=../../../../lib/${ARCH64}/libkstat.so.1
844 link path=usr/lib/${ARCH64}/libkvm.so target=libkvm.so.1
845 link path=usr/lib/${ARCH64}/libl.so target=libl.so.1
846 link path=usr/lib/${ARCH64}/libldap.so target=libldap.so.5
847 link path=usr/lib/${ARCH64}/liblddbg.so.4 \
848     target=../../../../lib/${ARCH64}/liblddbg.so.4
849 link path=usr/lib/${ARCH64}/liblgrp.so target=liblgrp.so.1
850 link path=usr/lib/${ARCH64}/liblm.so target=liblm.so.1
851 link path=usr/lib/${ARCH64}/libmail.so target=libmail.so.1
852 link path=usr/lib/${ARCH64}/libmalloc.so target=libmalloc.so.1

```

```

853 link path=usr/lib/${ARCH64}/libmapmalloc.so target=libmapmalloc.so.1
854 link path=usr/lib/${ARCH64}/libmd.so target=../../../../lib/${ARCH64}/libmd.so.1
855 link path=usr/lib/${ARCH64}/libmd.so.1 \
856     target=../../../../lib/${ARCH64}/libmd.so.1
857 link path=usr/lib/${ARCH64}/libmd5.so \
858     target=../../../../lib/${ARCH64}/libmd5.so.1
859 link path=usr/lib/${ARCH64}/libmd5.so.1 \
860     target=../../../../lib/${ARCH64}/libmd5.so.1
861 link path=usr/lib/${ARCH64}/libmenu.so target=libmenu.so.1
862 link path=usr/lib/${ARCH64}/libmp.so target=../../../../lib/${ARCH64}/libmp.so.2
863 link path=usr/lib/${ARCH64}/libmp.so.2 \
864     target=../../../../lib/${ARCH64}/libmp.so.2
865 link path=usr/lib/${ARCH64}/libmtmalloc.so target=libmtmalloc.so.1
866 link path=usr/lib/${ARCH64}/libnls.so target=libnls.so.1
867 link path=usr/lib/${ARCH64}/libnsl.so \
868     target=../../../../lib/${ARCH64}/libnsl.so.1
869 link path=usr/lib/${ARCH64}/libnsl.so.1 \
870     target=../../../../lib/${ARCH64}/libnsl.so.1
871 link path=usr/lib/${ARCH64}/libnvpair.so \
872     target=../../../../lib/${ARCH64}/libnvpair.so.1
873 link path=usr/lib/${ARCH64}/libnvpair.so.1 \
874     target=../../../../lib/${ARCH64}/libnvpair.so.1
875 link path=usr/lib/${ARCH64}/libpam.so \
876     target=../../../../lib/${ARCH64}/libpam.so.1
877 link path=usr/lib/${ARCH64}/libpam.so.1 \
878     target=../../../../lib/${ARCH64}/libpam.so.1
879 link path=usr/lib/${ARCH64}/libpanel.so target=libpanel.so.1
880 link path=usr/lib/${ARCH64}/libpkcs11.so target=libpkcs11.so.1
881 link path=usr/lib/${ARCH64}/libposix4.so \
882     target=../../../../lib/${ARCH64}/librt.so.1
883 link path=usr/lib/${ARCH64}/libposix4.so.1 \
884     target=../../../../lib/${ARCH64}/librt.so.1
885 link path=usr/lib/${ARCH64}/libproc.so \
886     target=../../../../lib/${ARCH64}/libproc.so.1
887 link path=usr/lib/${ARCH64}/libproc.so.1 \
888     target=../../../../lib/${ARCH64}/libproc.so.1
889 link path=usr/lib/${ARCH64}/libproject.so target=libproject.so.1
890 link path=usr/lib/${ARCH64}/libpthread.so \
891     target=../../../../lib/${ARCH64}/libpthread.so.1
892 link path=usr/lib/${ARCH64}/libpthread.so.1 \
893     target=../../../../lib/${ARCH64}/libpthread.so.1
894 link path=usr/lib/${ARCH64}/librcm.so \
895     target=../../../../lib/${ARCH64}/librcm.so.1
896 link path=usr/lib/${ARCH64}/librcm.so.1 \
897     target=../../../../lib/${ARCH64}/librcm.so.1
898 link path=usr/lib/${ARCH64}/librepase.so target=librepase.so.1
899 link path=usr/lib/${ARCH64}/libresolv.so \
900     target=../../../../lib/${ARCH64}/libresolv.so.2
901 link path=usr/lib/${ARCH64}/libresolv.so.2 \
902     target=../../../../lib/${ARCH64}/libresolv.so.2
903 ${i386_ONLY}link path=usr/lib/${ARCH64}/librestart.so \
904     target=../../../../lib/${ARCH64}/librestart.so.1
905 link path=usr/lib/${ARCH64}/librestart.so.1 \
906     target=../../../../lib/${ARCH64}/librestart.so.1
907 link path=usr/lib/${ARCH64}/librpcsvc.so \
908     target=../../../../lib/${ARCH64}/librpcsvc.so.1
909 link path=usr/lib/${ARCH64}/librpcsvc.so.1 \
910     target=../../../../lib/${ARCH64}/librpcsvc.so.1
911 link path=usr/lib/${ARCH64}/librt.so target=../../../../lib/${ARCH64}/librt.so.1
912 link path=usr/lib/${ARCH64}/librt.so.1 \
913     target=../../../../lib/${ARCH64}/librt.so.1
914 link path=usr/lib/${ARCH64}/librtld.so.1 \
915     target=../../../../lib/${ARCH64}/librtld.so.1
916 link path=usr/lib/${ARCH64}/librtld_db.so \
917     target=../../../../lib/${ARCH64}/librtld_db.so.1
918 link path=usr/lib/${ARCH64}/librtld_db.so.1 \

```

```

919 target=../../../../lib/${ARCH64}/librtld_db.so.1
920 link path=usr/lib/${ARCH64}/libsfcf.so \
921 target=../../../../lib/${ARCH64}/libsfcf.so.1
922 link path=usr/lib/${ARCH64}/libsfcf.so.1 \
923 target=../../../../lib/${ARCH64}/libsfcf.so.1
924 link path=usr/lib/${ARCH64}/libsched.so target=libsched.so.1
925 link path=usr/lib/${ARCH64}/libsctp.so target=libsctp.so.1
926 link path=usr/lib/${ARCH64}/libsec.so \
927 target=../../../../lib/${ARCH64}/libsec.so.1
928 link path=usr/lib/${ARCH64}/libsec.so.1 \
929 target=../../../../lib/${ARCH64}/libsec.so.1
930 link path=usr/lib/${ARCH64}/libsecdb.so \
931 target=../../../../lib/${ARCH64}/libsecdb.so.1
932 link path=usr/lib/${ARCH64}/libsecdb.so.1 \
933 target=../../../../lib/${ARCH64}/libsecdb.so.1
934 link path=usr/lib/${ARCH64}/libsndfile.so \
935 target=../../../../lib/${ARCH64}/libsndfile.so.1
936 link path=usr/lib/${ARCH64}/libsndfile.so.1 \
937 target=../../../../lib/${ARCH64}/libsndfile.so.1
938 link path=usr/lib/${ARCH64}/libsip.so target=libsip.so.1
939 link path=usr/lib/${ARCH64}/libslldap.so target=libslldap.so.1
940 link path=usr/lib/${ARCH64}/libsmbios.so target=libsmbios.so.1
941 link path=usr/lib/${ARCH64}/libsocket.so \
942 target=../../../../lib/${ARCH64}/libsocket.so.1
943 link path=usr/lib/${ARCH64}/libsocket.so.1 \
944 target=../../../../lib/${ARCH64}/libsocket.so.1
945 link path=usr/lib/${ARCH64}/libsoftcrypto.so target=libsoftcrypto.so.1
946 link path=usr/lib/${ARCH64}/libsysevent.so \
947 target=../../../../lib/${ARCH64}/libsysevent.so.1
948 link path=usr/lib/${ARCH64}/libsysevent.so.1 \
949 target=../../../../lib/${ARCH64}/libsysevent.so.1
950 link path=usr/lib/${ARCH64}/libtermcap.so \
951 target=../../../../lib/${ARCH64}/libtermcap.so.1
952 link path=usr/lib/${ARCH64}/libtermcap.so.1 \
953 target=../../../../lib/${ARCH64}/libtermcap.so.1
954 link path=usr/lib/${ARCH64}/libtermplib.so \
955 target=../../../../lib/${ARCH64}/libcurses.so.1
956 link path=usr/lib/${ARCH64}/libtermplib.so.1 \
957 target=../../../../lib/${ARCH64}/libcurses.so.1
958 link path=usr/lib/${ARCH64}/libthread.so \
959 target=../../../../lib/${ARCH64}/libthread.so.1
960 link path=usr/lib/${ARCH64}/libthread.so.1 \
961 target=../../../../lib/${ARCH64}/libthread.so.1
962 link path=usr/lib/${ARCH64}/libthread_db.so \
963 target=../../../../lib/${ARCH64}/libc_db.so.1
964 link path=usr/lib/${ARCH64}/libthread_db.so.1 \
965 target=../../../../lib/${ARCH64}/libc_db.so.1
966 link path=usr/lib/${ARCH64}/libtsnet.so \
967 target=../../../../lib/${ARCH64}/libtsnet.so.1
968 link path=usr/lib/${ARCH64}/libtsnet.so.1 \
969 target=../../../../lib/${ARCH64}/libtsnet.so.1
970 link path=usr/lib/${ARCH64}/libtsol.so \
971 target=../../../../lib/${ARCH64}/libtsol.so.2
972 link path=usr/lib/${ARCH64}/libtsol.so.2 \
973 target=../../../../lib/${ARCH64}/libtsol.so.2
974 link path=usr/lib/${ARCH64}/libumem.so \
975 target=../../../../lib/${ARCH64}/libumem.so.1
976 link path=usr/lib/${ARCH64}/libumem.so.1 \
977 target=../../../../lib/${ARCH64}/libumem.so.1
978 link path=usr/lib/${ARCH64}/libuuid.so \
979 target=../../../../lib/${ARCH64}/libuuid.so.1
980 link path=usr/lib/${ARCH64}/libuuid.so.1 \
981 target=../../../../lib/${ARCH64}/libuuid.so.1
982 $(i386_ONLY)link path=usr/lib/${ARCH64}/libuutil.so \
983 target=../../../../lib/${ARCH64}/libuutil.so.1
984 link path=usr/lib/${ARCH64}/libuutil.so.1 \

```

```

985 target=../../../../lib/${ARCH64}/libuutil.so.1
986 $(sparc_ONLY)link path=usr/lib/${ARCH64}/libv12n.so target=libv12n.so.1
987 link path=usr/lib/${ARCH64}/libvolmgt.so target=libvolmgt.so.1
988 link path=usr/lib/${ARCH64}/libw.so target=../../../../lib/${ARCH64}/libw.so.1
989 link path=usr/lib/${ARCH64}/libw.so.1 target=../../../../lib/${ARCH64}/libw.so.1
990 link path=usr/lib/${ARCH64}/libxnet.so \
991 target=../../../../lib/${ARCH64}/libxnet.so.1
992 link path=usr/lib/${ARCH64}/libxnet.so.1 \
993 target=../../../../lib/${ARCH64}/libxnet.so.1
994 link path=usr/lib/${ARCH64}/liby.so target=liby.so.1
995 link path=usr/lib/${ARCH64}/libzoneinfo.so target=libzoneinfo.so.1
996 link path=usr/lib/${ARCH64}/nss_compat.so.1 \
997 target=../../../../lib/${ARCH64}/nss_compat.so.1
998 link path=usr/lib/${ARCH64}/nss_dns.so.1 \
999 target=../../../../lib/${ARCH64}/nss_dns.so.1
1000 link path=usr/lib/${ARCH64}/nss_files.so.1 \
1001 target=../../../../lib/${ARCH64}/nss_files.so.1
1002 link path=usr/lib/${ARCH64}/nss_nis.so.1 \
1003 target=../../../../lib/${ARCH64}/nss_nis.so.1
1004 link path=usr/lib/${ARCH64}/nss_user.so.1 \
1005 target=../../../../lib/${ARCH64}/nss_user.so.1
1006 link path=usr/lib/${ARCH64}/straddr.so target=straddr.so.2
1007 link path=usr/lib/32 target=.
1008 link path=usr/lib/64 target=${ARCH64}
1009 link path=usr/lib/cfgadm/${ARCH64}/ib.so target=ib.so.1
1010 link path=usr/lib/cfgadm/${ARCH64}/pci.so target=pci.so.1
1011 $(i386_ONLY)link path=usr/lib/cfgadm/${ARCH64}/sata.so target=sata.so.1
1012 link path=usr/lib/cfgadm/${ARCH64}/scsi.so target=scsi.so.1
1013 link path=usr/lib/cfgadm/${ARCH64}/shp.so target=shp.so.1
1014 link path=usr/lib/cfgadm/${ARCH64}/usb.so target=usb.so.1
1015 link path=usr/lib/cfgadm/ib.so target=ib.so.1
1016 link path=usr/lib/cfgadm/pci.so target=pci.so.1
1017 $(i386_ONLY)link path=usr/lib/cfgadm/sata.so target=sata.so.1
1018 link path=usr/lib/cfgadm/scsi.so target=scsi.so.1
1019 link path=usr/lib/cfgadm/shp.so target=shp.so.1
1020 link path=usr/lib/cfgadm/usb.so target=usb.so.1
1021 link path=usr/lib/libadm.so target=libadm.so.1
1022 link path=usr/lib/libadm.so.1 target=libadm.so.1
1023 link path=usr/lib/libadutils.so target=libadutils.so.1
1024 link path=usr/lib/libaio.so target=libaio.so.1
1025 link path=usr/lib/libaio.so.1 target=libaio.so.1
1026 link path=usr/lib/libavl.so.1 target=libavl.so.1
1027 link path=usr/lib/libbsdmalloc.so target=libbsdmalloc.so.1
1028 link path=usr/lib/libbsm.so target=libbsm.so.1
1029 link path=usr/lib/libbsm.so.1 target=libbsm.so.1
1030 link path=usr/lib/libc.so target=libc.so.1
1031 link path=usr/lib/libc.so.1 target=libc.so.1
1032 link path=usr/lib/libc_db.so target=libc_db.so.1
1033 link path=usr/lib/libc_db.so.1 target=libc_db.so.1
1034 link path=usr/lib/libcfgadm.so target=libcfgadm.so.1
1035 link path=usr/lib/libcmd.so target=libcmd.so.1
1036 link path=usr/lib/libcmdutils.so.1 target=libcmdutils.so.1
1037 link path=usr/lib/libcommputil.so target=libcommputil.so.1
1038 link path=usr/lib/libcontract.so target=libcontract.so.1
1039 link path=usr/lib/libcontract.so.1 target=libcontract.so.1
1040 link path=usr/lib/libcrypt.so target=libcrypt.so.1
1041 link path=usr/lib/libcrypt_d.so target=libcrypt.so.1
1042 link path=usr/lib/libcrypt_d.so.1 target=libcrypt.so.1
1043 link path=usr/lib/libcrypt_i.so target=libcrypt.so.1
1044 link path=usr/lib/libcrypt_i.so.1 target=libcrypt.so.1
1045 link path=usr/lib/libctf.so target=libctf.so.1
1046 link path=usr/lib/libctf.so.1 target=libctf.so.1
1047 link path=usr/lib/libcurses.so target=libcurses.so.1
1048 link path=usr/lib/libcurses.so.1 target=libcurses.so.1
1049 link path=usr/lib/libdevice.so target=libdevice.so.1
1050 link path=usr/lib/libdevice.so.1 target=libdevice.so.1

```

```

1051 link path=usr/lib/libdevid.so target=../lib/libdevid.so.1
1052 link path=usr/lib/libdevid.so.1 target=../lib/libdevid.so.1
1053 link path=usr/lib/libdevinfo.so target=../lib/libdevinfo.so.1
1054 link path=usr/lib/libdevinfo.so.1 target=../lib/libdevinfo.so.1
1055 link path=usr/lib/libdhcpcagent.so.1 target=../lib/libdhcpcagent.so.1
1056 link path=usr/lib/libdhcputil.so.1 target=../lib/libdhcputil.so.1
1057 link path=usr/lib/libdisasm.so target=../lib/libdisasm.so.1
1058 link path=usr/lib/libdl.so target=../lib/libdl.so.1
1059 link path=usr/lib/libdl.so.1 target=../lib/libdl.so.1
1060 link path=usr/lib/libdlopen.so target=../lib/libdlopen.so.1
1061 link path=usr/lib/libdlopen.so.1 target=../lib/libdlopen.so.1
1062 link path=usr/lib/libdoor.so target=../lib/libdoor.so.1
1063 link path=usr/lib/libdoor.so.1 target=../lib/libdoor.so.1
1064 link path=usr/lib/libefi.so target=../lib/libefi.so.1
1065 link path=usr/lib/libefi.so.1 target=../lib/libefi.so.1
1066 link path=usr/lib/libelf.so target=../lib/libelf.so.1
1067 link path=usr/lib/libelf.so.1 target=../lib/libelf.so.1
1068 link path=usr/lib/libexacct.so target=../lib/libexacct.so.1
1069 $(i386_ONLY)link path=usr/lib/libfdisk.so target=../lib/libfdisk.so.1
1070 $(i386_ONLY)link path=usr/lib/libfdisk.so.1 target=../lib/libfdisk.so.1
1071 link path=usr/lib/libform.so target=../lib/libform.so.1
1072 link path=usr/lib/libfstyp.so target=../lib/libfstyp.so.1
1073 link path=usr/lib/libgen.so target=../lib/libgen.so.1
1074 link path=usr/lib/libgen.so.1 target=../lib/libgen.so.1
1075 link path=usr/lib/libhotplug.so target=../lib/libhotplug.so.1
1076 link path=usr/lib/libidmap.so target=../lib/libidmap.so.1
1077 link path=usr/lib/libinetutil.so.1 target=../lib/libinetutil.so.1
1078 link path=usr/lib/libintl.so target=../lib/libintl.so.1
1079 link path=usr/lib/libintl.so.1 target=../lib/libintl.so.1
1080 link path=usr/lib/libipmi.so target=../lib/libipmi.so.1
1081 link path=usr/lib/libipp.so target=../lib/libipp.so.1
1082 link path=usr/lib/libkstat.so target=../lib/libkstat.so.1
1083 link path=usr/lib/libkstat.so.1 target=../lib/libkstat.so.1
1084 link path=usr/lib/libkvm.so target=../lib/libkvm.so.1
1085 link path=usr/lib/libl.so target=../lib/libl.so.1
1086 link path=usr/lib/libldap.so target=../lib/libldap.so.1
1087 link path=usr/lib/liblddbg.so.4 target=../lib/liblddbg.so.4
1088 link path=usr/lib/liblgrp.so target=../lib/liblgrp.so.1
1089 link path=usr/lib/liblm.so target=../lib/liblm.so.1
1090 link path=usr/lib/libmail.so target=../lib/libmail.so.1
1091 link path=usr/lib/libmalloc.so target=../lib/libmalloc.so.1
1092 link path=usr/lib/libmapmalloc.so target=../lib/libmapmalloc.so.1
1093 link path=usr/lib/libmd.so target=../lib/libmd.so.1
1094 link path=usr/lib/libmd.so.1 target=../lib/libmd.so.1
1095 link path=usr/lib/libmd5.so target=../lib/libmd5.so.1
1096 link path=usr/lib/libmd5.so.1 target=../lib/libmd5.so.1
1097 link path=usr/lib/libmenu.so target=../lib/libmenu.so.1
1098 link path=usr/lib/libmp.so target=../lib/libmp.so.1
1099 link path=usr/lib/libmp.so.1 target=../lib/libmp.so.1
1100 link path=usr/lib/libmp.so.2 target=../lib/libmp.so.2
1101 link path=usr/lib/libmtdm.so target=../lib/libmtdm.so.1
1102 link path=usr/lib/libnls.so target=../lib/libnls.so.1
1103 link path=usr/lib/libnls.so target=../lib/libnls.so.1
1104 link path=usr/lib/libnls.so.1 target=../lib/libnls.so.1
1105 link path=usr/lib/libnvpair.so target=../lib/libnvpair.so.1
1106 link path=usr/lib/libnvpair.so.1 target=../lib/libnvpair.so.1
1107 link path=usr/lib/libpam.so target=../lib/libpam.so.1
1108 link path=usr/lib/libpam.so.1 target=../lib/libpam.so.1
1109 link path=usr/lib/libpanel.so target=../lib/libpanel.so.1
1110 link path=usr/lib/libpkcs11.so target=../lib/libpkcs11.so.1
1111 link path=usr/lib/libposix4.so target=../lib/librt.so.1
1112 link path=usr/lib/libposix4.so.1 target=../lib/librt.so.1
1113 link path=usr/lib/libproc.so target=../lib/libproc.so.1
1114 link path=usr/lib/libproc.so.1 target=../lib/libproc.so.1
1115 link path=usr/lib/libproject.so target=../lib/libproject.so.1
1116 link path=usr/lib/libpthread.so target=../lib/libpthread.so.1

```

```

1117 link path=usr/lib/libpthread.so.1 target=../lib/libpthread.so.1
1118 link path=usr/lib/librcm.so target=../lib/librcm.so.1
1119 link path=usr/lib/librcm.so.1 target=../lib/librcm.so.1
1120 link path=usr/lib/libreparse.so target=../lib/libreparse.so.1
1121 link path=usr/lib/libresolv.so target=../lib/libresolv.so.2
1122 link path=usr/lib/libresolv.so.1 target=../lib/libresolv.so.1
1123 link path=usr/lib/libresolv.so.2 target=../lib/libresolv.so.2
1124 link path=usr/lib/librestart.so.1 target=../lib/librestart.so.1
1125 link path=usr/lib/librpcsvc.so target=../lib/librpcsvc.so.1
1126 link path=usr/lib/librpcsvc.so.1 target=../lib/librpcsvc.so.1
1127 link path=usr/lib/librt.so target=../lib/librt.so.1
1128 link path=usr/lib/librt.so.1 target=../lib/librt.so.1
1129 link path=usr/lib/librtld.so.1 target=../lib/librtld.so.1
1130 link path=usr/lib/librtld_db.so target=../lib/librtld_db.so.1
1131 link path=usr/lib/librtld_db.so.1 target=../lib/librtld_db.so.1
1132 link path=usr/lib/librtld_db.so.1 target=../lib/librtld_db.so.1
1133 link path=usr/lib/libscf.so.1 target=../lib/libscf.so.1
1134 link path=usr/lib/libscf.so.1 target=../lib/libscf.so.1
1135 link path=usr/lib/libscf.so.1 target=../lib/libscf.so.1
1136 link path=usr/lib/libsec.so target=../lib/libsec.so.1
1137 link path=usr/lib/libsec.so.1 target=../lib/libsec.so.1
1138 link path=usr/lib/libsecdb.so target=../lib/libsecdb.so.1
1139 link path=usr/lib/libsecdb.so.1 target=../lib/libsecdb.so.1
1140 link path=usr/lib/libsendfile.so target=../lib/libsendfile.so.1
1141 link path=usr/lib/libsendfile.so.1 target=../lib/libsendfile.so.1
1142 link path=usr/lib/libsip.so target=../lib/libsip.so.1
1143 link path=usr/lib/libldap.so target=../lib/libldap.so.1
1144 link path=usr/lib/libmbios.so target=../lib/libmbios.so.1
1145 link path=usr/lib/libsocket.so target=../lib/libsocket.so.1
1146 link path=usr/lib/libsocket.so.1 target=../lib/libsocket.so.1
1147 link path=usr/lib/libsoftcrypto.so target=../lib/libsoftcrypto.so.1
1148 link path=usr/lib/libsys.so target=../lib/libsys.so.1
1149 link path=usr/lib/libsysevent.so target=../lib/libsysevent.so.1
1150 link path=usr/lib/libsysevent.so.1 target=../lib/libsysevent.so.1
1151 link path=usr/lib/libtermcap.so target=../lib/libtermcap.so.1
1152 link path=usr/lib/libtermcap.so.1 target=../lib/libtermcap.so.1
1153 link path=usr/lib/libtermmlib.so target=../lib/libcurses.so.1
1154 link path=usr/lib/libtermmlib.so.1 target=../lib/libcurses.so.1
1155 link path=usr/lib/libthread.so target=../lib/libthread.so.1
1156 link path=usr/lib/libthread.so.1 target=../lib/libthread.so.1
1157 link path=usr/lib/libthread_db.so target=../lib/libc_db.so.1
1158 link path=usr/lib/libthread_db.so.1 target=../lib/libc_db.so.1
1159 link path=usr/lib/libtsnet.so target=../lib/libtsnet.so.1
1160 link path=usr/lib/libtsnet.so.1 target=../lib/libtsnet.so.1
1161 link path=usr/lib/libtsol.so target=../lib/libtsol.so.2
1162 link path=usr/lib/libtsol.so.2 target=../lib/libtsol.so.2
1163 link path=usr/lib/libumem.so target=../lib/libumem.so.1
1164 link path=usr/lib/libumem.so.1 target=../lib/libumem.so.1
1165 link path=usr/lib/libuuid.so target=../lib/libuuid.so.1
1166 link path=usr/lib/libuuid.so.1 target=../lib/libuuid.so.1
1167 link path=usr/lib/libutil.so.1 target=../lib/libutil.so.1
1168 $(sparc_ONLY)link path=usr/lib/libv12n.so target=../lib/libv12n.so.1
1169 link path=usr/lib/libvolmgt.so target=../lib/libvolmgt.so.1
1170 link path=usr/lib/libw.so target=../lib/libw.so.1
1171 link path=usr/lib/libw.so.1 target=../lib/libw.so.1
1172 link path=usr/lib/libwrap.so target=../lib/libwrap.so.1.0
1173 link path=usr/lib/libwrap.so.1 target=../lib/libwrap.so.1.0
1174 link path=usr/lib/libxnet.so target=../lib/libxnet.so.1
1175 link path=usr/lib/libxnet.so.1 target=../lib/libxnet.so.1
1176 link path=usr/lib/liby.so target=../lib/liby.so.1
1177 link path=usr/lib/libzoneinfo.so target=../lib/libzoneinfo.so.1
1178 link path=usr/lib/lwp/$(ARCH64)/libthread.so.1 \
target=../lib/$(ARCH64)/libthread.so.1
1179 link path=usr/lib/lwp/$(ARCH64)/libthread_db.so.1 \
target=../lib/$(ARCH64)/libthread_db.so.1
1181 link path=usr/lib/lwp/32 target=../lib/libw.so.1
1182 link path=usr/lib/lwp/32 target=../lib/libw.so.1

```

```

1183 link path=usr/lib/lwp/64 target=$(ARCH64)
1184 link path=usr/lib/lwp/libthread.so.1 target=../libthread.so.1
1185 link path=usr/lib/lwp/libthread_db.so.1 target=../libthread_db.so.1
1186 link path=usr/lib/nss_compat.so.1 target=../lib/nss_compat.so.1
1187 link path=usr/lib/nss_dns.so.1 target=../lib/nss_dns.so.1
1188 link path=usr/lib/nss_files.so.1 target=../lib/nss_files.so.1
1189 link path=usr/lib/nss_nis.so.1 target=../lib/nss_nis.so.1
1190 link path=usr/lib/nss_user.so.1 target=../lib/nss_user.so.1
1191 link path=usr/lib/scsi/${ARCH64}/libscsi.so target=../libscsi.so.1
1192 link path=usr/lib/scsi/${ARCH64}/libsmp.so target=../libsmp.so.1
1193 link path=usr/lib/scsi/${ARCH64}/libsmp.so target=../libsmp.so.1
1194 link path=usr/lib/scsi/libscsi.so target=../libscsi.so.1
1195 link path=usr/lib/scsi/libses.so target=../libses.so.1
1196 link path=usr/lib/scsi/libscsi/libscsi.so target=../libscsi.so.1
1197 link path=usr/lib/security/${ARCH64}/crypt_bsdbf.so target=../crypt_bsdbf.so.1
1198 link path=usr/lib/security/${ARCH64}/crypt_bsdmd5.so \
1199 target=../crypt_bsdmd5.so.1
1200 link path=usr/lib/security/${ARCH64}/crypt_sha256.so \
1201 target=../crypt_sha256.so.1
1202 link path=usr/lib/security/${ARCH64}/crypt_sha512.so \
1203 target=../crypt_sha512.so.1
1204 link path=usr/lib/security/${ARCH64}/crypt_sunmd5.so \
1205 target=../crypt_sunmd5.so.1
1206 link path=usr/lib/security/${ARCH64}/pam_allow.so target=../pam_allow.so.1
1207 link path=usr/lib/security/${ARCH64}/pam_authtok_check.so \
1208 target=../pam_authtok_check.so.1
1209 link path=usr/lib/security/${ARCH64}/pam_authtok_get.so \
1210 target=../pam_authtok_get.so.1
1211 link path=usr/lib/security/${ARCH64}/pam_authtok_store.so \
1212 target=../pam_authtok_store.so.1
1213 link path=usr/lib/security/${ARCH64}/pam_deny.so target=../pam_deny.so.1
1214 link path=usr/lib/security/${ARCH64}/pam_dhkeys.so target=../pam_dhkeys.so.1
1215 link path=usr/lib/security/${ARCH64}/pam_dial_auth.so \
1216 target=../pam_dial_auth.so.1
1217 link path=usr/lib/security/${ARCH64}/pam_ldap.so target=../pam_ldap.so.1
1218 link path=usr/lib/security/${ARCH64}/pam_list.so target=../pam_list.so.1
1219 link path=usr/lib/security/${ARCH64}/pam_passwd_auth.so \
1220 target=../pam_passwd_auth.so.1
1221 link path=usr/lib/security/${ARCH64}/pam_rhosts_auth.so \
1222 target=../pam_rhosts_auth.so.1
1223 link path=usr/lib/security/${ARCH64}/pam_roles.so target=../pam_roles.so.1
1224 link path=usr/lib/security/${ARCH64}/pam_sample.so target=../pam_sample.so.1
1225 link path=usr/lib/security/${ARCH64}/pam_tsol_account.so \
1226 target=../pam_tsol_account.so.1
1227 link path=usr/lib/security/${ARCH64}/pam_unix_account.so \
1228 target=../pam_unix_account.so.1
1229 link path=usr/lib/security/${ARCH64}/pam_unix_auth.so \
1230 target=../pam_unix_auth.so.1
1231 link path=usr/lib/security/${ARCH64}/pam_unix_cred.so \
1232 target=../pam_unix_cred.so.1
1233 link path=usr/lib/security/${ARCH64}/pam_unix_session.so \
1234 target=../pam_unix_session.so.1
1235 link path=usr/lib/security/${ARCH64}/pkcs11_kernel.so \
1236 target=../pkcs11_kernel.so.1
1237 link path=usr/lib/security/${ARCH64}/pkcs11_softtoken.so \
1238 target=../pkcs11_softtoken.so.1
1239 link path=usr/lib/security/${ARCH64}/pkcs11_tpm.so target=../pkcs11_tpm.so.1
1240 link path=usr/lib/security/64 target=$(ARCH64)
1241 link path=usr/lib/security/audit_binfile.so target=../audit_binfile.so.1
1242 link path=usr/lib/security/audit_remote.so target=../audit_remote.so.1
1243 link path=usr/lib/security/audit_syslog.so target=../audit_syslog.so.1
1244 link path=usr/lib/security/crypt_bsdbf.so target=../crypt_bsdbf.so.1
1245 link path=usr/lib/security/crypt_bsdmd5.so target=../crypt_bsdmd5.so.1
1246 link path=usr/lib/security/crypt_sha256.so target=../crypt_sha256.so.1
1247 link path=usr/lib/security/crypt_sha512.so target=../crypt_sha512.so.1
1248 link path=usr/lib/security/crypt_sunmd5.so target=../crypt_sunmd5.so.1

```

```

1249 link path=usr/lib/security/pam_allow.so target=../pam_allow.so.1
1250 link path=usr/lib/security/pam_authtok_check.so \
1251 target=../pam_authtok_check.so.1
1252 link path=usr/lib/security/pam_authtok_get.so target=../pam_authtok_get.so.1
1253 link path=usr/lib/security/pam_authtok_store.so \
1254 target=../pam_authtok_store.so.1
1255 link path=usr/lib/security/pam_deny.so target=../pam_deny.so.1
1256 link path=usr/lib/security/pam_dhkeys.so target=../pam_dhkeys.so.1
1257 link path=usr/lib/security/pam_dial_auth.so target=../pam_dial_auth.so.1
1258 link path=usr/lib/security/pam_ldap.so target=../pam_ldap.so.1
1259 link path=usr/lib/security/pam_list.so target=../pam_list.so.1
1260 link path=usr/lib/security/pam_passwd_auth.so target=../pam_passwd_auth.so.1
1261 link path=usr/lib/security/pam_rhosts_auth.so target=../pam_rhosts_auth.so.1
1262 link path=usr/lib/security/pam_roles.so target=../pam_roles.so.1
1263 link path=usr/lib/security/pam_sample.so target=../pam_sample.so.1
1264 link path=usr/lib/security/pam_tsol_account.so target=../pam_tsol_account.so.1
1265 link path=usr/lib/security/pam_unix_account.so target=../pam_unix_account.so.1
1266 link path=usr/lib/security/pam_unix_auth.so target=../pam_unix_auth.so.1
1267 link path=usr/lib/security/pam_unix_cred.so target=../pam_unix_cred.so.1
1268 link path=usr/lib/security/pam_unix_session.so target=../pam_unix_session.so.1
1269 link path=usr/lib/security/pkcs11_kernel.so target=../pkcs11_kernel.so.1
1270 link path=usr/lib/security/pkcs11_softtoken.so target=../pkcs11_softtoken.so.1
1271 link path=usr/lib/security/pkcs11_tpm.so target=../pkcs11_tpm.so.1
1272 link path=usr/lib/straddr.so target=../straddr.so.2
1273 link path=usr/xpg4/lib/${ARCH64}/libcurses.so target=libcurses.so.2
1274 link path=usr/xpg4/lib/64 target=$(ARCH64)
1275 link path=usr/xpg4/lib/libcurses.so target=../libcurses.so.2
1276 #
1277 # libses.so needs to dlopen(3C) plugins from usr/lib/scsi/plugins/ses/vendor/,
1278 # a dependency which cannot be automatically derived
1279 #
1280 depend fmri=system/library/storage/scsi-plugins type=require

```