

```

*****
26232 Mon Oct 7 22:03:09 2013
new/usr/src/cmd/truss/actions.c
4142 truss should expand connect() arguments
*****
_____unchanged_portion_omitted_____

339 #define ISREAD(code) \
340 ((code) == SYS_read || (code) == SYS_pread || \
341 (code) == SYS_pread64 || (code) == SYS_readv || \
342 (code) == SYS_recv || (code) == SYS_recvfrom)
343 #define ISWRITE(code) \
344 ((code) == SYS_write || (code) == SYS_pwrite || \
345 (code) == SYS_pwrite64 || (code) == SYS_writew || \
346 (code) == SYS_send || (code) == SYS_sendto)

348 /*
349  * Return TRUE iff syscall is being traced.
350  */
351 int
352 sysentry(private_t *pri, int dotrace)
353 {
354     pid_t pid = Pstatus(Proc)->pr_pid;
355     const lwpstatus_t *Lsp = pri->lwpstat;
356     long arg;
357     int nargs;
358     int i;
359     int x;
360     int len;
361     char *s;
362     const struct systable *stp;
363     int what = Lsp->pr_what;
364     int subcode;
365     int istraced;
366     int raw;

368     /* for reporting sleeping system calls */
369     if (what == 0 && (Lsp->pr_flags & (PR_ASLEEP|PR_VFORKP)))
370         what = Lsp->pr_syscall;

372     /* protect ourself from operating system error */
373     if (what <= 0 || what > PRMAXSYS)
374         what = 0;

376     /*
377      * Set up the system call arguments (pri->sys_nargs & pri->sys_args[]).
378      */
379     setupsysargs(pri, what);
380     nargs = pri->sys_nargs;

382     /* get systable entry for this syscall */
383     subcode = getsubcode(pri);
384     stp = subsys(what, subcode);

386     if (nargs > stp->nargs)
387         nargs = stp->nargs;
388     pri->sys_nargs = nargs;

390     /*
391      * Fetch and remember first argument if it's a string,
392      * or second argument if SYS_openat or SYS_openat64.
393      */
394     pri->sys_valid = FALSE;
395     if ((nargs > 0 && stp->arg[0] == STG) ||
396         (nargs > 1 && (what == SYS_openat || what == SYS_openat64))) {
397         long offset;

```

```

398     uint32_t offset32;

400     /*
401      * Special case for exit from exec().
402      * The address in pri->sys_args[0] refers to the old process
403      * image. We must fetch the string from the new image.
404      */
405     if (Lsp->pr_why == PR_SYSEXIT && what == SYS_execve) {
406         psinfo_t psinfo;
407         long argv;
408         auxv_t auxv[32];
409         int naux;

411         offset = 0;
412         naux = proc_get_auxv(pid, auxv, 32);
413         for (i = 0; i < naux; i++) {
414             if (auxv[i].a_type == AT_SUN_EXECNAME) {
415                 offset = (long)auxv[i].a_un.a_ptr;
416                 break;
417             }
418         }
419         if (offset == 0 &&
420             proc_get_psinfo(pid, &psinfo) == 0) {
421             argv = (long)psinfo.pr_argv;
422             if (data_model == PR_MODEL_LP64)
423                 (void) Pread(Proc, &offset,
424                             sizeof (offset), argv);
425             else {
426                 offset32 = 0;
427                 (void) Pread(Proc, &offset32,
428                             sizeof (offset32), argv);
429                 offset = offset32;
430             }
431         }
432     } else if (stp->arg[0] == STG) {
433         offset = pri->sys_args[0];
434     } else {
435         offset = pri->sys_args[1];
436     }
437     if ((s = fetchstring(pri, offset, PATH_MAX)) != NULL) {
438         pri->sys_valid = TRUE;
439         len = strlen(s);
440         /* reallocate if necessary */
441         while (len >= pri->sys_psize) {
442             free(pri->sys_path);
443             pri->sys_path = my_malloc(pri->sys_psize * 2,
444                                     "pathname buffer");
445         }
446         (void) strcpy(pri->sys_path, s); /* remember pathname */
447     }
448 }

450     istraced = dotrace && prismember(&trace, what);
451     raw = prismember(&rawout, what);

453     /* force tracing of read/write buffer dump syscalls */
454     if (!istraced && nargs > 2) {
455         int fdpl = (int)pri->sys_args[0] + 1;

457         if (ISREAD(what)) {
458             if (prismember(&readfd, fdpl))
459                 istraced = TRUE;
460         } else if (ISWRITE(what)) {
461             if (prismember(&writefd, fdpl))
462                 istraced = TRUE;
463         }

```

```
464     }
465
466     pri->sys_leng = 0;
467     if (cflag || !istraced)
468         *pri->sys_string = 0;
469     else {
470         int argprinted = FALSE;
471         const char *name;
472
473         name = sysname(pri, what, raw? -1 : subcode);
474         grow(pri, strlen(name) + 1);
475         pri->sys_leng = snprintf(pri->sys_string, pri->sys_ssize,
476                                "%s(", name);
477         for (i = 0; i < nargs; i++) {
478             arg = pri->sys_args[i];
479             x = stp->arg[i];
480
481             if (!raw && pri->sys_valid &&
482                 ((i == 0 && x == STG) ||
483                  (i == 1 && (what == SYS_openat ||
484                             what == SYS_openat64)))) { /* already fetched */
485                 if (argprinted)
486                     outstring(pri, ", ");
487                 escape_string(pri, pri->sys_path);
488                 argprinted = TRUE;
489             } else if (x != NOV && (x != HID || raw)) {
490                 if (argprinted)
491                     outstring(pri, ", ");
492                 if (x == LLO || x == SAD)
493                     (*Print[x])(pri, raw, arg,
494                                pri->sys_args[++i]);
495                 else
496                     (*Print[x])(pri, raw, arg);
497                 argprinted = TRUE;
498             }
499         }
500         outstring(pri, ")");
501     }
502
503     return (istraced);
504 }
505 unchanged_portion_omitted
```

new/usr/src/cmd/truss/print.c

1

```
*****
69738 Mon Oct 7 22:03:10 2013
new/usr/src/cmd/truss/print.c
4142 truss should expand connect() arguments
*****
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) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
24 */

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

29 /* Copyright (c) 2013, OmniTI Computer Consulting, Inc. All rights reserved. */

31 #define _SYSCALL32      /* make 32-bit compat headers visible */

33 #include <stdio.h>
34 #include <stdlib.h>
35 #include <unistd.h>
36 #include <string.h>
37 #include <signal.h>
38 #include <termio.h>
39 #include <stddef.h>
40 #include <limits.h>
41 #include <fcntl.h>
42 #include <ctype.h>
43 #include <sys/types.h>
44 #include <sys/mman.h>
45 #include <sys/resource.h>
46 #include <sys/ulimit.h>
47 #include <sys/utsname.h>
48 #include <sys/kstat.h>
49 #include <sys/modctl.h>
50 #include <sys/acl.h>
51 #include <stropts.h>
52 #include <sys/isa_defs.h>
53 #include <sys/systeminfo.h>
54 #include <sys/cladm.h>
55 #include <sys/lwp.h>
56 #include <bsm/audit.h>
57 #include <libproc.h>
58 #include <priv.h>
59 #include <sys/aio.h>
60 #include <sys/aiocb.h>
61 #include <sys/corectl.h>
```

new/usr/src/cmd/truss/print.c

2

```
62 #include <sys/cpc_impl.h>
63 #include <sys/prioctl.h>
64 #include <sys/tsprioctl.h>
65 #include <sys/iaprioctl.h>
66 #include <sys/rtpriocntl.h>
67 #include <sys/fssprioctl.h>
68 #include <sys/fixprioctl.h>
69 #include <netdb.h>
70 #include <nss_dbdefs.h>
71 #include <sys/socketvar.h>
72 #include <netinet/in.h>
73 #include <netinet/tcp.h>
74 #include <netinet/udp.h>
75 #include <netinet/sctp.h>
76 #include <net/route.h>
77 #include <sys/utrap.h>
78 #include <sys/lgrp_user.h>
79 #include <sys/door.h>
80 #include <sys/tsol/tndb.h>
81 #include <sys/rctl.h>
82 #include <sys/rctl_impl.h>
83 #include <sys/fork.h>
84 #include <sys/task.h>
85 #include <sys/socket.h>
86 #include <arpa/inet.h>
87 #include "ramdata.h"
88 #include "print.h"
89 #include "proto.h"
90 #include "systable.h"

92 void grow(private_t *, int nbyte);

94 #define GROW(nb) if (pri->sys_leng + (nb) >= pri->sys_ssize) grow(pri, (nb))

97 /*ARGSUSED*/
98 void
99 prt_nov(private_t *pri, int raw, long val)      /* print nothing */
100 {
101 }
102
103 _____
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170 /*
171  * Print connect() 2nd and 3rd arguments.
172  */
173 /*ARGSUSED*/
174 void
175 prt_sad(private_t *pri, int raw, long addr, long len)
176 {
177     /*
178      * A buffer large enough for PATH_MAX size AF_UNIX address, which is
179      * also large enough to store a sockaddr_in or a sockaddr_in6.
180      */
181     long buf[(sizeof (short) + PATH_MAX + sizeof (long) - 1) /
182              sizeof (long)];
183     struct sockaddr *sa = (struct sockaddr *)buf;
184     struct sockaddr_in *sin = (struct sockaddr_in *)buf;
185     struct sockaddr_un *soun = (struct sockaddr_un *)buf;
186     struct sockaddr_in6 *sin6 = (struct sockaddr_in6 *)buf;
187     char addrbuf[INET6_ADDRSTRLEN];
188     char scope[8];
189     long rlen = len;

1811     GROW(20);
1812     if (data_model == PR_MODEL_ILP32) {
1813         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
```

```

1814         "0x%.8X", (int)addr);
1815     } else {
1816         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1817             "0x%.8lX", addr);
1818     }
1820     if (raw != 0)
1821         return;
1823     if (rlen >= sizeof (buf)) /* protect against ridiculous length */
1824         rlen = sizeof (buf) - 1;
1825     if (Pread(Proc, buf, rlen, addr) != rlen)
1826         return;
1828     GROW(175);
1830     switch (sa->sa_family) {
1831     case AF_INET6:
1832         if (ntohl(sin6->sin6_scope_id) == 0) {
1833             scope[0] = '\0';
1834         } else {
1835             sprintf(scope, "%%u", ntohl(sin6->sin6_scope_id));
1836         }
1837         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1838             " AF_INET6 to = %s%s:%u",
1839             inet_ntop(AF_INET6, &sin6->sin6_addr, addrbuf,
1840                 sizeof (addrbuf)),
1841             scope, ntohs(sin6->sin6_port));
1842         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1843             " source id = 0x%x",
1844             " flow class = 0x%02x flow label = 0x%05x",
1845             ntohl(sin6->sin6_src_id),
1846             ntohl((sin6->sin6_flowinfo & IPV6_FLOWINFO_TCLASS) >> 20),
1847             ntohl(sin6->sin6_flowinfo & IPV6_FLOWINFO_FLOWLABEL));
1848         break;
1849     case AF_INET:
1850         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1851             " AF_INET to = %s:%u",
1852             inet_ntop(AF_INET, &sin->sin_addr, addrbuf,
1853                 sizeof (addrbuf)),
1854             ntohs(sin->sin_port));
1855         break;
1856     case AF_UNIX:
1857         len -= sizeof (soun->sun_family);
1858         if (len >= 0) {
1859             /* Null terminate */
1860             soun->sun_path[len] = NULL;
1861             pri->sys_leng += sprintf(
1862                 pri->sys_string + pri->sys_leng,
1863                 " AF_UNIX to = %s", soun->sun_path);
1864         }
1865         break;
1866     }
1868     /*
1869     * print the third argument len
1870     */
1871     if (data_model == PR_MODEL_ILP32) {
1872         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1873             ", %d", (int)len);
1874     } else {
1875         pri->sys_leng += sprintf(pri->sys_string + pri->sys_leng,
1876             ", %ld", len);
1877     }
1879 }

```

```

1882 /*
1883 * Print accept4() flags argument.
1884 */
1885 void
1886 prt_acf(private_t *pri, int raw, long val)
1887 {
1888     int first = 1;
1889     if (raw || !val ||
1890         (val & ~(SOCK_CLOEXEC|SOCK_NDELAY|SOCK_NONBLOCK))) {
1891         prt_dex(pri, 0, val);
1892         return;
1893     }
1895     if (val & SOCK_CLOEXEC) {
1896         outstring(pri, "|SOCK_CLOEXEC" + first);
1897         first = 0;
1898     }
1899     if (val & SOCK_NDELAY) {
1900         outstring(pri, "|SOCK_NDELAY" + first);
1901         first = 0;
1902     }
1903     if (val & SOCK_NONBLOCK) {
1904         outstring(pri, "|SOCK_NONBLOCK" + first);
1905     }
1906 }
1907 unchanged_portion_omitted
2838 /*
2839 * Array of pointers to print functions, one for each format.
2840 */
2841 void (* const Print[])() = {
2842     prt_nov, /* NOV -- no value */
2843     prt_dec, /* DEC -- print value in decimal */
2844     prt_oct, /* OCT -- print value in octal */
2845     prt_hex, /* HEX -- print value in hexadecimal */
2846     prt_dex, /* DEX -- print value in hexadecimal if big enough */
2847     prt_stg, /* STG -- print value as string */
2848     prt_ioc, /* IOC -- print ioctl code */
2849     prt_fcn, /* FCN -- print fcntl code */
2850     prt_s86, /* S86 -- print sysi86 code */
2851     prt_uts, /* UTS -- print utssys code */
2852     prt_opn, /* OPN -- print open code */
2853     prt_sig, /* SIG -- print signal name plus flags */
2854     prt_uat, /* UAT -- print unlinkat() flag */
2855     prt_msc, /* MSC -- print msgsys command */
2856     prt_msf, /* MSF -- print msgsys flags */
2857     prt_smc, /* SMC -- print semsys command */
2858     prt_sef, /* SEF -- print semsys flags */
2859     prt_shc, /* SHC -- print shmsys command */
2860     prt_shf, /* SHF -- print shmsys flags */
2861     prt_fat, /* FAT -- print faccessat( flag */
2862     prt_sfs, /* SFS -- print sysfs code */
2863     prt_rst, /* RST -- print string returned by syscall */
2864     prt_smf, /* SMF -- print streams message flags */
2865     prt_ioa, /* IOA -- print ioctl argument */
2866     prt_pip, /* PIP -- print pipe flags */
2867     prt_mtf, /* MTF -- print mount flags */
2868     prt_mft, /* MFT -- print mount file system type */
2869     prt_iob, /* IOB -- print contents of I/O buffer */
2870     prt_hhx, /* HHX -- print value in hexadecimal (half size) */
2871     prt_wop, /* WOP -- print waitsys() options */
2872     prt_spm, /* SPM -- print sigprocmask argument */
2873     prt_rlk, /* RLK -- print readlink buffer */
2874     prt_mpr, /* MPR -- print mmap()/mprotect() flags */

```

```

2875     prt_mty,      /* MTY -- print mmap() mapping type flags */
2876     prt_mcf,      /* MCF -- print memcntl() function */
2877     prt_mc4,      /* MC4 -- print memcntl() (fourth) argument */
2878     prt_mc5,      /* MC5 -- print memcntl() (fifth) argument */
2879     prt_mad,      /* MAD -- print madvise() argument */
2880     prt_ulm,      /* ULM -- print ulimit() argument */
2881     prt_rlm,      /* RLM -- print get/setrlimit() argument */
2882     prt_cnf,      /* CNF -- print sysconfig() argument */
2883     prt_inf,      /* INF -- print sysinfo() argument */
2884     prt_ptc,      /* PTC -- print pathconf/fpathconf() argument */
2885     prt_fui,      /* FUI -- print fusers() input argument */
2886     prt_idt,      /* IDT -- print idtype_t, waitid() argument */
2887     prt_lwf,      /* LWF -- print lwp_create() flags */
2888     prt_itm,      /* ITM -- print [get|set]itimer() arg */
2889     prt_llo,      /* LLO -- print long long offset arg */
2890     prt_mod,      /* MOD -- print modctl() subcode */
2891     prt_wnh,      /* WHN -- print lseek() whence argument */
2892     prt_acl,      /* ACL -- print acl() code */
2893     prt_aio,      /* AIO -- print kaio() code */
2894     prt_aud,      /* AUD -- print auditsys() code */
2895     prt_uns,      /* DEC -- print value in unsigned decimal */
2896     prt_clc,      /* CLC -- print cladm command argument */
2897     prt_clf,      /* CLF -- print cladm flag argument */
2898     prt_cor,      /* COR -- print corectl() subcode */
2899     prt_cco,      /* CCO -- print corectl() options */
2900     prt_ccc,      /* CCC -- print corectl() content */
2901     prt_rcc,      /* RCC -- print corectl() returned content */
2902     prt_cpc,      /* CPC -- print cpc() subcode */
2903     prt_sqc,      /* SQC -- print sigqueue() si_code argument */
2904     prt_pc4,      /* PC4 -- print pricntlsys() (fourth) argument */
2905     prt_pc5,      /* PC5 -- print pricntlsys() (key, value) pairs */
2906     prt_pst,      /* PST -- print processor set id */
2907     prt_mif,      /* MIF -- print meminfo() arguments */
2908     prt_pfm,      /* PFM -- print so_socket() proto-family (1st) arg */
2909     prt_skt,      /* SKT -- print so_socket() socket-type (2nd) arg */
2910     prt_skp,      /* SKP -- print so_socket() protocol (3rd) arg */
2911     prt_skv,      /* SKV -- print socket version arg */
2912     prt_sad,      /* SAD -- print connect 2nd and 3rd arguments */
2913     prt_sol,      /* SOL -- print [sg]setsockopt() level (2nd) arg */
2914     prt_son,      /* SON -- print [sg]setsockopt() opt-name (3rd) arg */
2915     prt_utt,      /* UTT -- print utrap type */
2916     prt_uth,      /* UTH -- print utrap handler */
2917     prt_acc,      /* ACC -- print access() flags */
2918     prt_sht,      /* SHT -- print shutdown() how (2nd) argument */
2919     prt_ffg,      /* FFG -- print fcntl() flags (3rd) argument */
2920     prt_prs,      /* PRS -- print privilege set */
2921     prt_pro,      /* PRO -- print privilege set operation */
2922     prt_prn,      /* PRN -- print privilege set name */
2923     prt_pfl,      /* PFL -- print privilege/process flag name */
2924     prt_laf,      /* LAF -- print lgrp_affinity arguments */
2925     prt_key,      /* KEY -- print key_t 0 as IPC_PRIVATE */
2926     prt_zga,      /* ZGA -- print zone_getattr attribute types */
2927     prt_atc,      /* ATC -- print AT_FDCWD or file descriptor */
2928     prt_lio,      /* LIO -- print LIO_XX flags */
2929     prt_dfl,      /* DFL -- print door_create() flags */
2930     prt_dpm,      /* DPM -- print DOOR_PARAM_XX flags */
2931     prt_tnd,      /* TND -- print trusted network data base opcode */
2932     prt_rsc,      /* RSC -- print rctlsys() subcodes */
2933     prt_rgf,      /* RGF -- print getrctl() flags */
2934     prt_rsf,      /* RSF -- print setrctl() flags */
2935     prt_rcf,      /* RCF -- print rctlsys_ctl() flags */
2936     prt_fxf,      /* FXF -- print forkx() flags */
2937     prt_spf,      /* SPF -- print rctlsys_projset() flags */
2938     prt_unl,      /* UNL -- as prt_uns except for -1 */
2939     prt_mob,      /* MOB -- print mmapobj() flags */
2940     prt_snf,      /* SNF -- print AT_SYMLINK_[NO]FOLLOW flag */

```

```

2941     prt_skc,      /* SKC -- print sockconfig() subcode */
2942     prt_acf,      /* ACF -- print accept4 flags */
2943     prt_pfd,      /* PFD -- print pipe fds */
2944     prt_dec,      /* HID -- hidden argument, make this the last one */
2945 };
_____unchanged_portion_omitted_____

```

new/usr/src/cmd/truss/print.h

1

```
*****
6327 Mon Oct 7 22:03:10 2013
new/usr/src/cmd/truss/print.h
4142 truss should expand connect() arguments
*****
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) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
24 */

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

29 /* Copyright (c) 2013, OmniTI Computer Consulting, Inc. All right reserved. */

31 #ifndef _TRUSS_PRINT_H
32 #define _TRUSS_PRINT_H

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

38 /*
39  * Argument & return value print codes.
40 */
41 #define NOV      0          /* no value */
42 #define DEC      1          /* print value in decimal */
43 #define OCT      2          /* print value in octal */
44 #define HEX      3          /* print value in hexadecimal */
45 #define DEX      4          /* print value in hexadecimal if big enough */
46 #define STG      5          /* print value as string */
47 #define IOC      6          /* print ioctl code */
48 #define FCN      7          /* print fcntl code */
49 #define S86      8          /* print sysi86 code */
50 #define UTS      9          /* print utssys code */
51 #define OPN     10          /* print open code */
52 #define SIG     11          /* print signal name plus flags */
53 #define UAT     12          /* print unlinkat() flag */
54 #define MSC     13          /* print msgsys command */
55 #define MSF     14          /* print msgsys flags */
56 #define SMC     15          /* print semsys command */
57 #define SEF     16          /* print semsys flags */
58 #define SHC     17          /* print shmsys command */
59 #define SHF     18          /* print shmsys flags */
60 #define FAT     19          /* print faccessat() flag */
61 #define SFS     20          /* print sysfs code */
```

new/usr/src/cmd/truss/print.h

2

```
62 #define RST      21          /* print string returned by sys call */
63 #define SMF      22          /* print streams message flags */
64 #define IOA      23          /* print ioctl argument */
65 #define PIP      24          /* print pipe flags */
66 #define MTF      25          /* print mount flags */
67 #define MFT      26          /* print mount file system type */
68 #define IOB      27          /* print contents of I/O buffer */
69 #define HHX      28          /* print value in hexadecimal (half size) */
70 #define WOP      29          /* print waitsys() options */
71 #define SPM      30          /* print sigprocmask argument */
72 #define RLK      31          /* print readlink buffer */
73 #define MPR      32          /* print mmap()/mprotect() flags */
74 #define MTY      33          /* print mmap() mapping type flags */
75 #define MCF      34          /* print memcntl() function */
76 #define MC4      35          /* print memcntl() (fourth) argument */
77 #define MC5      36          /* print memcntl() (fifth) argument */
78 #define MAD      37          /* print madvise() argument */
79 #define ULM      38          /* print ulimit() argument */
80 #define RLM      39          /* print get/setrlimit() argument */
81 #define CNF      40          /* print sysconfig() argument */
82 #define INF      41          /* print sysinfo() argument */
83 #define PTC      42          /* print pathconf/fpathconf() argument */
84 #define FUI      43          /* print fusers() input argument */
85 #define IDT      44          /* print idtype_t, waitid() argument */
86 #define LWF      45          /* print lwp_create() flags */
87 #define ITM      46          /* print [get|set]itimer() arg */
88 #define LLO      47          /* print long long offset */
89 #define MOD      48          /* print modctl() code */
90 #define WHN      49          /* print lseek() whence argument */
91 #define ACL      50          /* print acl() code */
92 #define ATO      51          /* print kaio() code */
93 #define AUD      52          /* print auditsys() code */
94 #define UNS      53          /* print value in unsigned decimal */
95 #define CLC      54          /* print cladm() command argument */
96 #define CLF      55          /* print cladm() flag argument */
97 #define COR      56          /* print corectl() subcode */
98 #define CCO      57          /* print corectl() options */
99 #define CCC      58          /* print corectl() content */
100 #define RCC      59          /* print corectl() content */
101 #define CPC      60          /* print cpc() subcode */
102 #define SQC      61          /* print sigqueue() si_code argument */
103 #define PC4      62          /* print pricntlsys() (fourth) argument */
104 #define PC5      63          /* print pricntlsys() (key-value) pairs */
105 #define PST      64          /* print processor set id */
106 #define MIF      65          /* print meminfo() argument */
107 #define PFM      66          /* print so_socket() proto-family (1st) arg */
108 #define SKT      67          /* print so_socket() socket type (2nd) arg */
109 #define SKP      68          /* print so_socket() protocol (3rd) arg */
110 #define SKV      69          /* print so_socket() version (5th) arg */
111 #define SAD      70          /* print sockaddr and len for connect() */
112 #define SOL      71          /* print [sg]setsockopt() level (2nd) arg */
113 #define SON      72          /* print [sg]setsockopt() name (3rd) arg */
114 #define UTT      73          /* print utrap type */
115 #define UTH      74          /* print utrap handler */
116 #define ACC      75          /* print access flags */
117 #define SHT      76          /* print shutdown() "how" (2nd) arg */
118 #define FFG      77          /* print fcntl() flags (3rd) arg */
119 #define PRS      78          /* privilege set */
120 #define PRO      79          /* privilege set operation */
121 #define PRN      80          /* privilege set name */
122 #define PPL      81          /* privilege/process flag name */
123 #define LAF      82          /* print lgrp affinity arguments */
124 #define KEY      83          /* print key_t 0 as IPC_PRIVATE */
125 #define ZGA      84          /* print zone_getattr attribute types */
126 #define ATC      85          /* print AT_FDCWD or file descriptor */
127 #define LIO      86          /* print LIO_XX flags */
```

```

128 #define DFL      87      /* print door_create() flags */
129 #define DPM      88      /* print DOOR_PARAM_XX flags */
130 #define TND      89      /* print trusted network data base opcode */
131 #define RSC      90      /* print rctlsys subcode */
132 #define RGF      91      /* print rctlsys_get flags */
133 #define RSF      92      /* print rctlsys_set flags */
134 #define RCF      93      /* print rctlsys_ctl flags */
135 #define FXF      94      /* print forkx flags */
136 #define SPF      95      /* print rctlsys_projset flags */
137 #define UNL      96      /* unsigned except for -1 */
138 #define MOB      97      /* print mmapobj() flags */
139 #define SNF      98      /* print AT_SYMLINK_[NO]FOLLOW flag */
140 #define SKC      99      /* print sockconfig subcode */
141 #define ACF      100     /* accept4 flags */
142 #define PFD      101     /* pipe fds[2] */
143 #define HID      102     /* hidden argument, don't print */
111 #define SOL      70      /* print [sg]setsockopt() level (2nd) arg */
112 #define SON      71      /* print [sg]setsockopt() name (3rd) arg */
113 #define UTT      72      /* print utrap type */
114 #define UTH      73      /* print utrap handler */
115 #define ACC      74      /* print access flags */
116 #define SHT      75      /* print shutdown() "how" (2nd) arg */
117 #define FFG      76      /* print fcntl() flags (3rd) arg */
118 #define PRS      77      /* privilege set */
119 #define PRO      78      /* privilege set operation */
120 #define PRN      79      /* privilege set name */
121 #define PFL      80      /* privilege/process flag name */
122 #define LAF      81      /* print lgrp_affinity arguments */
123 #define KEY      82      /* print key_t 0 as IPC_PRIVATE */
124 #define ZGA      83      /* print zone_getattr attribute types */
125 #define ATC      84      /* print AT_FDCWD or file descriptor */
126 #define LIO      85      /* print LIO_XX flags */
127 #define DFL      86      /* print door_create() flags */
128 #define DPM      87      /* print DOOR_PARAM_XX flags */
129 #define TND      88      /* print trusted network data base opcode */
130 #define RSC      89      /* print rctlsys subcode */
131 #define RGF      90      /* print rctlsys_get flags */
132 #define RSF      91      /* print rctlsys_set flags */
133 #define RCF      92      /* print rctlsys_ctl flags */
134 #define FXF      93      /* print forkx flags */
135 #define SPF      94      /* print rctlsys_projset flags */
136 #define UNL      95      /* unsigned except for -1 */
137 #define MOB      96      /* print mmapobj() flags */
138 #define SNF      97      /* print AT_SYMLINK_[NO]FOLLOW flag */
139 #define SKC      98      /* print sockconfig subcode */
140 #define ACF      99      /* accept4 flags */
141 #define PFD      100     /* pipe fds[2] */
142 #define HID      101     /* hidden argument, don't print */
144                                     /* make sure HID is always the last member */

146 /*
147  * Print routines, indexed by print codes.
148  */
149 extern void (* const Print[])();

151 #ifdef __cplusplus
152 }
    unchanged_portion_omitted

```

```

*****
58001 Mon Oct 7 22:03:10 2013
new/usr/src/cmd/truss/systable.c
4142 src should expand connect() arguments
*****
_____unchanged_portion_omitted_____

```

```

220 const struct systable systable[] = {
221 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
222 {"_exit", 1, DEC, NOV, DEC}, /* 1 */
223 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
224 {"read", 3, DEC, NOV, DEC, IOB, UNS}, /* 3 */
225 {"write", 3, DEC, NOV, DEC, IOB, UNS}, /* 4 */
226 {"open", 3, DEC, NOV, STG, OPN, OCT}, /* 5 */
227 {"close", 1, DEC, NOV, DEC}, /* 6 */
228 {"linkat", 5, DEC, NOV, ATC, STG, ATC, STG, SNF}, /* 7 */
229 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
230 {"link", 2, DEC, NOV, STG, STG}, /* 9 */
231 {"unlink", 1, DEC, NOV, STG}, /* 10 */
232 {"symlinkat", 3, DEC, NOV, STG, ATC, STG}, /* 11 */
233 {"chdir", 1, DEC, NOV, STG}, /* 12 */
234 {"time", 0, DEC, NOV}, /* 13 */
235 {"mknod", 3, DEC, NOV, STG, OCT, HEX}, /* 14 */
236 {"chmod", 2, DEC, NOV, STG, OCT}, /* 15 */
237 {"chown", 3, DEC, NOV, STG, DEC, DEC}, /* 16 */
238 {"brk", 1, DEC, NOV, HEX}, /* 17 */
239 {"stat", 2, DEC, NOV, STG, HEX}, /* 18 */
240 {"lseek", 3, DEC, NOV, DEC, DEX, WHN}, /* 19 */
241 {"getpid", 0, DEC, DEC}, /* 20 */
242 {"mount", 8, DEC, NOV, STG, STG, MTF, MFT, HEX, DEC, HEX, DEC}, /* 21 */
243 {"readlinkat", 4, DEC, NOV, ATC, STG, RLK, UNS}, /* 22 */
244 {"setuid", 1, DEC, NOV, UNS}, /* 23 */
245 {"getuid", 0, UNS, UNS}, /* 24 */
246 {"stime", 1, DEC, NOV, DEC}, /* 25 */
247 {"pcsample", 2, DEC, NOV, HEX, DEC}, /* 26 */
248 {"alarm", 1, DEC, NOV, UNS}, /* 27 */
249 {"fstat", 2, DEC, NOV, DEC, HEX}, /* 28 */
250 {"pause", 0, DEC, NOV}, /* 29 */
251 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
252 {"stty", 2, DEC, NOV, DEC, DEC}, /* 31 */
253 {"gtty", 2, DEC, NOV, DEC, DEC}, /* 32 */
254 {"access", 2, DEC, NOV, STG, ACC}, /* 33 */
255 {"nice", 1, DEC, NOV, DEC}, /* 34 */
256 {"statfs", 4, DEC, NOV, STG, HEX, DEC, DEC}, /* 35 */
257 {"sync", 0, DEC, NOV}, /* 36 */
258 {"kill", 2, DEC, NOV, DEC, SIG}, /* 37 */
259 {"fstatfs", 4, DEC, NOV, DEC, HEX, DEC, DEC}, /* 38 */
260 {"pgrpsys", 3, DEC, NOV, DEC, DEC, DEC}, /* 39 */
261 {"uucopystr", 3, DEC, NOV, STG, RST, UNS}, /* 40 */
262 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
263 {"pipe", 2, DEC, NOV, PFD, PIP}, /* 42 */
264 {"times", 1, DEC, NOV, HEX}, /* 43 */
265 {"profil", 4, DEC, NOV, HEX, UNS, HEX, OCT}, /* 44 */
266 {"faccessat", 4, DEC, NOV, ATC, STG, ACC, FAT}, /* 45 */
267 {"setgid", 1, DEC, NOV, UNS}, /* 46 */
268 {"getgid", 0, UNS, UNS}, /* 47 */
269 {"mknodat", 4, DEC, NOV, ATC, STG, OCT, HEX}, /* 48 */
270 {"msgsys", 6, DEC, NOV, DEC, DEC, DEC, DEC, DEC}, /* 49 */
271 {"sysi86", 4, HEX, NOV, S86, HEX, HEX, HEX, DEC, DEC}, /* 50 */
272 {"acct", 1, DEC, NOV, STG}, /* 51 */
273 {"shmsys", 4, DEC, NOV, DEC, HEX, HEX, HEX}, /* 52 */
274 {"semsys", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX}, /* 53 */
275 {"ioctl", 3, DEC, NOV, DEC, IOC, IOA}, /* 54 */
276 {"uadmin", 3, DEC, NOV, DEC, DEC, DEC}, /* 55 */
277 {"fchownat", 5, DEC, NOV, ATC, STG, DEC, DEC, SNF}, /* 56 */

```

```

278 {"utssys", 4, DEC, NOV, HEX, DEC, UTS, HEX}, /* 57 */
279 {"fdsync", 2, DEC, NOV, DEC, FFG}, /* 58 */
280 {"execve", 3, DEC, NOV, STG, HEX, HEX}, /* 59 */
281 {"umask", 1, OCT, NOV, OCT}, /* 60 */
282 {"chroot", 1, DEC, NOV, STG}, /* 61 */
283 {"fcntl", 3, DEC, NOV, DEC, FCN, HEX}, /* 62 */
284 {"ulimit", 2, DEX, NOV, ULM, DEC}, /* 63 */
285 {"renameat", 4, DEC, NOV, ATC, STG, ATC, STG}, /* 64 */
286 {"unlinkat", 3, DEC, NOV, ATC, STG, UAT}, /* 65 */
287 {"fstatat", 4, DEC, NOV, ATC, STG, HEX, SNF}, /* 66 */
288 {"fstatat64", 4, DEC, NOV, ATC, STG, HEX, SNF}, /* 67 */
289 {"openat", 4, DEC, NOV, ATC, STG, OPN, OCT}, /* 68 */
290 {"openat64", 4, DEC, NOV, ATC, STG, OPN, OCT}, /* 69 */
291 {"tasksys", 5, DEC, NOV, DEC, DEC, DEC, HEX, DEC}, /* 70 */
292 {"acctctl", 3, DEC, NOV, HEX, HEX, UNS}, /* 71 */
293 {"exacctsys", 6, DEC, NOV, DEC, IDT, DEC, HEX, DEC, HEX}, /* 72 */
294 {"getpagesizes", 2, DEC, NOV, HEX, DEC}, /* 73 */
295 {"rctlsys", 6, DEC, NOV, RSC, STG, HEX, HEX, DEC, DEC}, /* 74 */
296 {"sidsys", 4, UNS, UNS, DEC, DEC, DEC, DEC}, /* 75 */
297 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
298 {"lwp_park", 3, DEC, NOV, DEC, HEX, DEC}, /* 77 */
299 {"sendfilev", 5, DEC, NOV, DEC, DEC, HEX, DEC, HEX}, /* 78 */
300 {"rmdir", 1, DEC, NOV, STG}, /* 79 */
301 {"mkdir", 2, DEC, NOV, STG, OCT}, /* 80 */
302 {"getdents", 3, DEC, NOV, DEC, HEX, UNS}, /* 81 */
303 {"privsys", 5, HEX, NOV, DEC, DEC, DEC, HEX, DEC}, /* 82 */
304 {"ucredsys", 3, DEC, NOV, DEC, DEC, HEX}, /* 83 */
305 {"sysfs", 3, DEC, NOV, SFS, DEX, DEX}, /* 84 */
306 {"getmsg", 4, DEC, NOV, DEC, HEX, HEX, HEX}, /* 85 */
307 {"putmsg", 4, DEC, NOV, DEC, HEX, HEX, SMF}, /* 86 */
308 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
309 {"lstat", 2, DEC, NOV, STG, HEX}, /* 88 */
310 {"symlink", 2, DEC, NOV, STG, STG}, /* 89 */
311 {"readlink", 3, DEC, NOV, STG, RLK, UNS}, /* 90 */
312 {"setgroups", 2, DEC, NOV, DEC, HEX}, /* 91 */
313 {"getgroups", 2, DEC, NOV, DEC, HEX}, /* 92 */
314 {"fchmod", 2, DEC, NOV, DEC, OCT}, /* 93 */
315 {"fchown", 3, DEC, NOV, DEC, DEC, DEC}, /* 94 */
316 {"sigprocmask", 3, DEC, NOV, SPM, HEX, HEX}, /* 95 */
317 {"sigsuspend", 1, DEC, NOV, HEX}, /* 96 */
318 {"sigaltstack", 2, DEC, NOV, HEX, HEX}, /* 97 */
319 {"sigaction", 3, DEC, NOV, SIG, HEX, HEX}, /* 98 */
320 {"sigpendsys", 2, DEC, NOV, DEC, HEX}, /* 99 */
321 {"context", 2, DEC, NOV, DEC, HEX}, /* 100 */
322 {"fchmodat", 4, DEC, NOV, ATC, STG, OCT, SNF}, /* 101 */
323 {"mkdirat", 3, DEC, NOV, ATC, STG, OCT}, /* 102 */
324 {"statvfs", 2, DEC, NOV, STG, HEX}, /* 103 */
325 {"fstatvfs", 2, DEC, NOV, DEC, HEX}, /* 104 */
326 {"getloadavg", 2, DEC, NOV, HEX, DEC}, /* 105 */
327 {"nfssys", 2, DEC, NOV, DEC, HEX}, /* 106 */
328 {"waitid", 4, DEC, NOV, IDT, DEC, HEX, WOP}, /* 107 */
329 {"sigsendsys", 2, DEC, NOV, HEX, SIG}, /* 108 */
330 {"hrtsys", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX}, /* 109 */
331 {"utimesys", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX}, /* 110 */
332 {"sigresend", 3, DEC, NOV, SIG, HEX, HEX}, /* 111 */
333 {"prioctlsys", 5, DEC, NOV, DEC, HEX, DEC, PC4, PC5}, /* 112 */
334 {"pathconf", 2, DEC, NOV, STG, PTC}, /* 113 */
335 {"mincore", 3, DEC, NOV, HEX, UNS, HEX}, /* 114 */
336 {"mmap", 6, HEX, NOV, HEX, UNS, MPR, MTY, DEC, DEC}, /* 115 */
337 {"mprotect", 3, DEC, NOV, HEX, UNS, MPR}, /* 116 */
338 {"munmap", 2, DEC, NOV, HEX, UNS}, /* 117 */
339 {"fpathconf", 2, DEC, NOV, DEC, PTC}, /* 118 */
340 {"vfork", 0, DEC, NOV}, /* 119 */
341 {"fchdir", 1, DEC, NOV, DEC}, /* 120 */
342 {"readv", 3, DEC, NOV, DEC, HEX, DEC}, /* 121 */
343 {"writev", 3, DEC, NOV, DEC, HEX, DEC}, /* 122 */

```



```

344 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
345 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
346 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
347 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
348 "mmapobj", 5, DEC, NOV, DEC, MOB, HEX, HEX, HEX }, /* 127 */
349 "setrlimit", 2, DEC, NOV, RLM, HEX }, /* 128 */
350 "getrlimit", 2, DEC, NOV, RLM, HEX }, /* 129 */
351 "lchown", 3, DEC, NOV, STG, DEC, DEC }, /* 130 */
352 "memcntl", 6, DEC, NOV, HEX, UNS, MCF, MC4, MC5, DEC }, /* 131 */
353 "getpmsg", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX }, /* 132 */
354 "putpmsg", 5, DEC, NOV, DEC, HEX, HEX, DEC, HXH }, /* 133 */
355 "rename", 2, DEC, NOV, STG, STG }, /* 134 */
356 "uname", 1, DEC, NOV, HEX }, /* 135 */
357 "setegid", 1, DEC, NOV, UNS }, /* 136 */
358 "sysconfig", 1, DEC, NOV, CNF }, /* 137 */
359 "adjtime", 2, DEC, NOV, HEX, HEX }, /* 138 */
360 "sysinfo", 3, DEC, NOV, INF, RST, DEC }, /* 139 */
361 "sharefs", 3, DEC, NOV, DEC, HEX, DEC }, /* 140 */
362 "seteuid", 1, DEC, NOV, UNS }, /* 141 */
363 "forksys", 2, DEC, NOV, DEC, HXH }, /* 142 */
364 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
365 "sigtimedwait", 3, DEC, NOV, HEX, HEX, HEX }, /* 144 */
366 "lwp_info", 1, DEC, NOV, HEX }, /* 145 */
367 "yield", 0, DEC, NOV }, /* 146 */
368 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
369 "lwp_sema_post", 1, DEC, NOV, HEX }, /* 148 */
370 "lwp_sema_trywait", 1, DEC, NOV, HEX }, /* 149 */
371 "lwp_detach", 1, DEC, NOV, DEC }, /* 150 */
372 "corectl", 4, DEC, NOV, DEC, HEX, HEX, HEX }, /* 151 */
373 "modctl", 5, DEC, NOV, MOD, HEX, HEX, HEX, HEX }, /* 152 */
374 "fchroot", 1, DEC, NOV, DEC }, /* 153 */
375 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
376 "vhangup", 0, DEC, NOV }, /* 155 */
377 "gettimeofday", 1, DEC, NOV, HEX }, /* 156 */
378 "getitimer", 2, DEC, NOV, ITM, HEX }, /* 157 */
379 "setitimer", 3, DEC, NOV, ITM, HEX, HEX }, /* 158 */
380 "lwp_create", 3, DEC, NOV, HEX, LWF, HEX }, /* 159 */
381 "lwp_exit", 0, DEC, NOV }, /* 160 */
382 "lwp_suspend", 1, DEC, NOV, DEC }, /* 161 */
383 "lwp_continue", 1, DEC, NOV, DEC }, /* 162 */
384 "lwp_kill", 2, DEC, NOV, DEC, SIG }, /* 163 */
385 "lwp_self", 0, DEC, NOV }, /* 164 */
386 "lwp_sigmask", 5, HEX, HEX, SPM, HEX, HEX, HEX, HEX }, /* 165 */
387 "lwp_private", 3, HEX, NOV, DEC, DEC, HEX }, /* 166 */
388 "lwp_wait", 2, DEC, NOV, DEC, HEX }, /* 167 */
389 "lwp_mutex_wakeup", 2, DEC, NOV, HEX, DEC }, /* 168 */
390 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
391 "lwp_cond_wait", 4, DEC, NOV, HEX, HEX, HEX, DEC }, /* 170 */
392 "lwp_cond_signal", 1, DEC, NOV, HEX }, /* 171 */
393 "lwp_cond_broadcast", 1, DEC, NOV, HEX }, /* 172 */
394 "pread", 4, DEC, NOV, DEC, IOB, UNS, DEX }, /* 173 */
395 "pwrite", 4, DEC, NOV, DEC, IOB, UNS, DEX }, /* 174 */
396 "llseek", 4, LLO, NOV, DEC, LLO, HID, WHN }, /* 175 */
397 "inst_sync", 2, DEC, NOV, STG, DEC }, /* 176 */
398 "brand", 6, DEC, NOV, DEC, HEX, HEX, HEX, HEX, HEX }, /* 177 */
399 "kaio", 7, DEC, NOV, AIO, HEX, HEX, HEX, HEX, HEX }, /* 178 */
400 "cpc", 5, DEC, NOV, CPC, DEC, HEX, HEX, HEX }, /* 179 */
401 "lgrpsys", 3, DEC, NOV, DEC, DEC, HEX }, /* 180 */
402 "rusagesys", 5, DEC, NOV, DEC, HEX, DEC, HEX, HEX }, /* 181 */
403 "ports", 6, HEX, NOV, DEC, HEX, HEX, HEX, HEX, HEX }, /* 182 */
404 "pollsys", 4, DEC, NOV, HEX, DEC, HEX, HEX }, /* 183 */
405 "labelsys", 2, DEC, NOV, DEC, HEX }, /* 184 */
406 "acl", 4, DEC, NOV, STG, ACL, DEC, HEX }, /* 185 */
407 "auditsys", 4, DEC, NOV, AUD, HEX, HEX, HEX }, /* 186 */
408 "processor_bind", 4, DEC, NOV, IDT, DEC, DEC, HEX }, /* 187 */
409 "processor_info", 2, DEC, NOV, DEC, HEX }, /* 188 */

```

```

410 {"p_online", 2, DEC, NOV, DEC, DEC }, /* 189 */
411 {"sigqueue", 5, DEC, NOV, DEC, SIG, HEX, SQC, DEC }, /* 190 */
412 "clock_gettime", 2, DEC, NOV, DEC, HEX }, /* 191 */
413 "clock_settime", 2, DEC, NOV, DEC, HEX }, /* 192 */
414 "clock_getres", 2, DEC, NOV, DEC, HEX }, /* 193 */
415 "timer_create", 3, DEC, NOV, DEC, HEX, HEX }, /* 194 */
416 "timer_delete", 1, DEC, NOV, DEC }, /* 195 */
417 "timer_settime", 4, DEC, NOV, DEC, DEC, HEX, HEX }, /* 196 */
418 "timer_gettime", 2, DEC, NOV, DEC, HEX }, /* 197 */
419 "timer_getoverrun", 1, DEC, NOV, DEC }, /* 198 */
420 "nanosleep", 2, DEC, NOV, HEX, HEX }, /* 199 */
421 "facl", 4, DEC, NOV, DEC, ACL, DEC, HEX }, /* 200 */
422 "door", 6, DEC, NOV, DEC, HEX, HEX, HEX, DEC }, /* 201 */
423 "setreuid", 2, DEC, NOV, UN1, UN1 }, /* 202 */
424 "setregid", 2, DEC, NOV, UN1, UN1 }, /* 203 */
425 "install_utrap", 3, DEC, NOV, DEC, HEX, HEX }, /* 204 */
426 "signotify", 3, DEC, NOV, DEC, HEX, HEX }, /* 205 */
427 "schedctl", 0, HEX, NOV }, /* 206 */
428 "pset", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX }, /* 207 */
429 "sparc_utrap_install", 5, DEC, NOV, UTT, UTH, UTH, HEX, HEX }, /* 208 */
430 "resolvepath", 3, DEC, NOV, STG, RLK, DEC }, /* 209 */
431 "lwp_mutex_timedlock", 3, DEC, NOV, HEX, HEX, HEX }, /* 210 */
432 "lwp_sema_timedwait", 3, DEC, NOV, HEX, HEX, DEC }, /* 211 */
433 "lwp_rwlock_sys", 3, DEC, NOV, DEC, HEX, HEX }, /* 212 */
434 "getdents64", 3, DEC, NOV, DEC, HEX, UNS }, /* 213 */
435 "mmap64", 7, HEX, NOV, HEX, UNS, MPR, MTY, DEC, LLO, HID }, /* 214 */
436 "stat64", 2, DEC, NOV, STG, HEX }, /* 215 */
437 "lstat64", 2, DEC, NOV, STG, HEX }, /* 216 */
438 "fstat64", 2, DEC, NOV, DEC, HEX }, /* 217 */
439 "statvfs64", 2, DEC, NOV, STG, HEX }, /* 218 */
440 "fstatvfs64", 2, DEC, NOV, DEC, HEX }, /* 219 */
441 "setrlimit64", 2, DEC, NOV, RLM, HEX }, /* 220 */
442 "getrlimit64", 2, DEC, NOV, RLM, HEX }, /* 221 */
443 "pread64", 5, DEC, NOV, DEC, IOB, UNS, LLO, HID }, /* 222 */
444 "pwrite64", 5, DEC, NOV, DEC, IOB, UNS, LLO, HID }, /* 223 */
445 { NULL, 8, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX, HEX },
446 "open64", 3, DEC, NOV, STG, OPN, OCT }, /* 225 */
447 "rpcmod", 3, DEC, NOV, DEC, HEX }, /* 226 */
448 "zone", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX }, /* 227 */
449 "autofssys", 2, DEC, NOV, DEC, HEX }, /* 228 */
450 "getcwd", 3, DEC, NOV, RST, DEC }, /* 229 */
451 "so_socket", 5, DEC, NOV, PFM, SKT, SKP, STG, SKV }, /* 230 */
452 "so_socketpair", 1, DEC, NOV, HEX }, /* 231 */
453 "bind", 4, DEC, NOV, DEC, HEX, DEC, SKV }, /* 232 */
454 "listen", 3, DEC, NOV, DEC, DEC, SKV }, /* 233 */
455 "accept", 5, DEC, NOV, DEC, HEX, HEX, SKV, ACF }, /* 234 */
456 {"connect", 4, DEC, NOV, DEC, SAD, DEC, SKV }, /* 235 */
456 {"connect", 4, DEC, NOV, DEC, HEX, DEC, SKV }, /* 235 */
457 "shutdown", 3, DEC, NOV, DEC, SHT, SKV }, /* 236 */
458 "recv", 4, DEC, NOV, DEC, IOB, DEC, DEC }, /* 237 */
459 "recvfrom", 6, DEC, NOV, DEC, IOB, DEC, DEC, HEX, HEX }, /* 238 */
460 "recvmsg", 3, DEC, NOV, DEC, HEX, DEC }, /* 239 */
461 "send", 4, DEC, NOV, DEC, IOB, DEC, DEC }, /* 240 */
462 "sendmsg", 3, DEC, NOV, DEC, HEX, DEC }, /* 241 */
463 "sendto", 6, DEC, NOV, DEC, IOB, DEC, DEC, HEX, DEC }, /* 242 */
464 "getpeername", 4, DEC, NOV, DEC, HEX, HEX, SKV }, /* 243 */
465 "getsockname", 4, DEC, NOV, DEC, HEX, HEX, SKV }, /* 244 */
466 "getsockopt", 6, DEC, NOV, DEC, SOL, SON, HEX, HEX, SKV }, /* 245 */
467 "setsockopt", 6, DEC, NOV, DEC, SOL, SON, HEX, DEC, SKV }, /* 246 */
468 "sockconfig", 5, DEC, NOV, DEC, HEX, HEX, HEX, HEX }, /* 247 */
469 "ntp_gettime", 1, DEC, NOV, HEX }, /* 248 */
470 "ntp_adjtime", 1, DEC, NOV, HEX }, /* 249 */
471 "lwp_mutex_unlock", 1, DEC, NOV, HEX }, /* 250 */
472 "lwp_mutex_trylock", 2, DEC, NOV, HEX, HEX }, /* 251 */
473 "lwp_mutex_register", 2, DEC, NOV, HEX, HEX }, /* 252 */
474 {"cladm", 3, DEC, NOV, CLC, CLF, HEX }, /* 253 */

```

new/usr/src/cmd/truss/systable.c

5

```
475 {"uucopy",      3, DEC, NOV, HEX, HEX, UNS}, /* 254 */
476 {"umount2",    2, DEC, NOV, STG, MTF}, /* 255 */
477 { NULL, -1, DEC, NOV},
478 };
_____unchanged_portion_omitted_____
```