```
**********************************************************
    8297 Thu Jan 24 09:54:37 2019
new/usr/src/cmd/fs.d/reparsed/reparsed.c
10133 smatch fixes for usr/src/cmd/fs.d
**********************************************************
```

```
   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 2009 Sun Microsystems, Inc.  All rights reserved.
  23  * Use is subject to license terms.
  24  */

  26 /*
  27  * Copyright (c) 2018, Joyent, Inc.
  28  */

  30 /*
  31  * Reparsed daemon
  32  */

  34 #include <stdio.h>
  35 #include <stdio_ext.h>
  36 #include <stdlib.h>
  37 #include <unistd.h>
  38 #include <signal.h>
  39 #include <sys/types.h>
  40 #include <sys/stat.h>
  41 #include <fcntl.h>
  42 #include <memory.h>
  43 #include <alloca.h>
  44 #include <ucontext.h>
  45 #include <errno.h>
  46 #include <syslog.h>
  47 #include <string.h>
  48 #include <strings.h>
  49 #include <door.h>
  50 #include <wait.h>
  51 #include <libintl.h>
  52 #include <locale.h>
  53 #include <sys/param.h>
  54 #include <sys/systeminfo.h>
  55 #include <sys/thread.h>
  56 #include <rpc/xdr.h>
  57 #include <priv.h>
  58 #include <sys/fs_reparse.h>
  59 #include <priv_utils.h>
  60 #include <rpcsvc/daemon_utils.h>
```



```
  62 #define REPARSED_CMD_OPTS       "v"
  63 #define DOOR_RESULT_BUFSZ       (MAXPATHLEN + sizeof (reparsed_door_res_t))
  64 #define SAFETY_BUFFER           8*1024

  66 static char *MyName;
  67 static int verbose = 0;

  69 static int start_reparsed_svcs();
  70 static void daemonize(void);
  71 static void reparsed_door_call_error(int error, int buflen);
  72 static void reparsed_doorfunc(void *cookie, char *argp, size_t arg_size,
  73                         door_desc_t *dp, uint_t n_desc);

  75 static void
  76 usage()
  77 {
  78         syslog(LOG_ERR, "Usage: %s", MyName);
  79         syslog(LOG_ERR, "\t[-v]\t\tverbose error messages)");
  80         exit(1);
  81 }
______unchanged_portion_omitted_

 190 static void
 191 reparsed_door_call_error(int error, int buflen)
 192 {
 193         reparsed_door_res_t rpd_res;

 195         memset(&rpd_res, 0, sizeof (reparsed_door_res_t));
 196         rpd_res.res_status = error;
 197         rpd_res.res_len = buflen;
 198         (void) door_return((char *)&rpd_res,
 199             sizeof (reparsed_door_res_t), NULL, 0);
 194         door_return((char *)&rpd_res, sizeof (reparsed_door_res_t), NULL, 0);

 201         (void) door_return(NULL, 0, NULL, 0);
 202         abort();
 197         /* NOTREACHED */
 203 }
______unchanged_portion_omitted_
```

```
**********************************************************
    4980 Thu Jan 24 09:54:37 2019
new/usr/src/cmd/fs.d/smbclnt/chacl/chacl.c
10133 smatch fixes for usr/src/cmd/fs.d
**********************************************************
```

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

/*
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*
 * Copyright (c) 2018, Joyent, Inc.
 */

/*
 * This is the smbfs/chacl command.
 * (just for testing - not installed)
 *
 * Works like chmod(1), but only supporting A=... forms.
 * i.e. chacl A=everyone@:full_set:fd:allow /mnt/foo
 *
 * Some more test cases:
 *      /usr/lib/fs/smbfs/chacl -v
 *      A=user:2147483649:rwxpdDaARWcCos::allow,
 *      user:2147483653:raRcs::allow,
 *      everyone@:raRcs::allow
 */

#include <sys/types.h>
#include <sys/errno.h>
#include <sys/stat.h>
#include <sys/acl.h>
#include <sys/acl_impl.h>

#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <aclutils.h>

#include <netsmb/smbfs_acl.h>

char *progname;
int Vflag;
```



```
void chacl(char *, uint32_t, uid_t, gid_t, acl_t *);

static const char Usage[] =
        "Usage: %s [-v] [-u UID] [-g GID] A=ACL... file ...\n"
        "\twhere A=ACL is like chmod(1)\n";

void
usage(void)
{
        fprintf(stderr, Usage, progname);
        exit(1);
}

int
main(int argc, char **argv)
{
        uid_t uid = (uid_t)-1;
        gid_t gid = (gid_t)-1;
        acl_t *acl = NULL;
        char *acl_arg;
        ulong_t tl;
        int c, error;
        uint32_t selector;

        progname = argv[0];

        while ((c = getopt(argc, argv, "vu:g:")) != -1) {
                switch (c) {
                case 'v':
                        Vflag++;
                        break;
                case 'u':
                        tl = strtoul(optarg, NULL, 10);
                        if (tl == 0)
                                goto badopt;
                        uid = (uid_t)tl;
                        break;
                case 'g':
                        tl = strtoul(optarg, NULL, 10);
                        if (tl == 0)
                                goto badopt;
                        gid = (gid_t)tl;
                        break;
                case ':':
                        fprintf(stderr, "%s: option %c requires arg\n",
                            progname, c);
                        usage();
                        break;

                badopt:
                default:
                        fprintf(stderr, "%s: bad option: %c\n",
                            progname, c);
                        usage();
                        break;
                }
        }

        if (optind + 1 > argc)
                usage();
        acl_arg = argv[optind++];

        /*
         * Ask libsec to parse the ACL arg.
         */
```

```
128         if (strncmp(acl_arg, "A=", 2) != 0)
129                 usage();
130         error = acl_parse(acl_arg + 2, &acl);
131         if (error) {
132                 fprintf(stderr, "%s: can not parse ACL: %s\n",
133                     progname, acl_arg);
134                 exit(1);
135         }
136         if (acl->acl_type != ACE_T) {
137                 fprintf(stderr, "%s: ACL not ACE_T type: %s\n",
138                     progname, acl_arg);
139                 exit(1);
140         }

142         /*
143          * Which parts of the SD are being modified?
144          */
145         selector = DACL_SECURITY_INFORMATION;

141         selector = 0;
142         if (acl)
143                 selector |= DACL_SECURITY_INFORMATION;
147         if (uid != (uid_t)-1)
148                 selector |= OWNER_SECURITY_INFORMATION;
149         if (gid != (gid_t)-1)
150                 selector |= GROUP_SECURITY_INFORMATION;

152         if (optind == argc)
153                 usage();
154         for (; optind < argc; optind++)
155                 chacl(argv[optind], selector, uid, gid, acl);

157 done:
158         acl_free(acl);
159         return (0);
160 }
```

*unchanged_portion_omitted_*

```
**********************************************************
    4162 Thu Jan 24 09:54:37 2019
new/usr/src/cmd/fs.d/smbclnt/umount/umount.c
10133 smatch fixes for usr/src/cmd/fs.d
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*
 * Copyright (c) 2018, Joyent, Inc.
 */

/*
 * smbfs umount
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdarg.h>
#include <signal.h>
#include <unistd.h>
#include <kstat.h>
#include <rpc/rpc.h>
#include <sys/mnttab.h>
#include <sys/mount.h>
#include <sys/mntent.h>
#include <errno.h>
#include <locale.h>
#include <fslib.h>
#include <priv_utils.h>

#define RET_OK  0
#define RET_ERR 32

static void pr_err(const char *fmt, ...);
static void usage();
static int smbfs_unmount(char *, int);
static struct extmnttab *mnttab_find();

static char *myname;
static char typename[64];

int
```



```
main(int argc, char *argv[])
{
        extern int optind;
        int c;
        int umnt_flag = 0;

        (void) setlocale(LC_ALL, "");

#if !defined(TEXT_DOMAIN)
#define TEXT_DOMAIN "SYS_TEST"
#endif
        (void) textdomain(TEXT_DOMAIN);

        /*
         * Normal users are allowed to umount smbfs mounts they own.
         * To allow that, this program is installed setuid root, and
         * it adds SYS_MOUNT privilege here (if needed), and then
         * restores the user's normal privileges.
         */
        if (__init_suid_priv(0, PRIV_SYS_MOUNT, (char *)NULL) < 0) {
                (void) fprintf(stderr,
                    gettext("Insufficient privileges, "
                    "%s must be set-uid root\n"), argv[0]);
                exit(RET_ERR);
        }

        myname = strrchr(argv[0], '/');
        myname = myname ? myname+1 : argv[0];
        (void) sprintf(typename, "smbfs %s", myname);
        argv[0] = typename;

        /*
         * Set options
         */
        while ((c = getopt(argc, argv, "f")) != EOF) {
                switch (c) {
                case 'f':
                        umnt_flag |= MS_FORCE; /* forced unmount is desired */
                        break;
                default:
                        usage();
                        exit(RET_ERR);
                }
        }
        if (argc - optind != 1) {
                usage();
                exit(RET_ERR);
        }

        return (smbfs_unmount(argv[optind], umnt_flag));
}
```

*_____unchanged_portion_omitted_*

```
/*
 *  Find the mnttab entry that corresponds to "name".
 *  We're not sure what the name represents: either
 *  a mountpoint name, or a special name (server:/path).
 *  Return the last entry in the file that matches.
 */
static struct extmnttab *
mnttab_find(dirname)
        char *dirname;
{
        FILE *fp;
        struct extmnttab mnt;
        struct extmnttab *res = NULL;
```

```
172         fp = fopen(MNTTAB, "r");
173         if (fp == NULL) {
174                 pr_err("%s: %s\n", MNTTAB, strerror(errno));
175                 return (NULL);
176         }
177         while (getextmntent(fp, &mnt, sizeof (struct extmnttab)) == 0) {
178                 if (strcmp(mnt.mnt_mountp, dirname) == 0 ||
179                     strcmp(mnt.mnt_special, dirname) == 0) {
180                         if (res)
181                                 fsfreemnttab(res);
182                         res = fsdupmnttab(&mnt);
183                 }
184         }

186         (void) fclose(fp);
182         fclose(fp);
187         return (res);
188 }
```

_____unchanged_portion_omitted_

```
**********************************************************
  161880 Thu Jan 24 09:54:38 2019
new/usr/src/cmd/fs.d/ufs/mkfs/mkfs.c
10133 smatch fixes for usr/src/cmd/fs.d
**********************************************************
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright (c) 1988, 2010, Oracle and/or its affiliates. All rights reserved.
 */

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

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

/*
 * Copyright (c) 2018, Joyent, Inc.
 */

/*
 * The maximum supported file system size (in sectors) is the
 * number of frags that can be represented in an int32_t field
 * (INT_MAX) times the maximum number of sectors per frag.  Since
 * the maximum frag size is MAXBSIZE, the maximum number of sectors
 * per frag is MAXBSIZE/DEV_BSIZE.
 */
#define FS_MAX  (((diskaddr_t)INT_MAX) * (MAXBSIZE/DEV_BSIZE))

/*
 * make file system for cylinder-group style file systems
 *
 * usage:
 *
 *    mkfs [-F FSType] [-V] [-G [-P]] [-M dirname] [-m] [options]
 *      [-o specific_options]  special size
 *      [nsect ntrack bsize fsize cpg   minfree rps nbpi opt apc rotdelay
 *        2     3      4     5     6    7       8   9    10  11  12
 *      nrpos maxcontig mtb]
```



```
 *      13    14        15
 *
 *  where specific_options are:
 *      N - no create
 *      nsect - The number of sectors per track
 *      ntrack - The number of tracks per cylinder
 *      bsize - block size
 *      fragsize - fragment size
 *      cgsize - The number of disk cylinders per cylinder group.
 *      free - minimum free space
 *      rps - rotational speed (rev/sec).
 *      nbpi - number of data bytes per allocated inode
 *      opt - optimization (space, time)
 *      apc - number of alternates
 *      gap - gap size
 *      nrpos - number of rotational positions
 *      maxcontig - maximum number of logical blocks that will be
 *              allocated contiguously before inserting rotational delay
 *      mtb - if "y", set up file system for eventual growth to over a
 *              a terabyte
 * -P Do not grow the file system, but print on stdout the maximal
 *    size in sectors to which the file system can be increased. The calculated
 *    size is limited by the value provided by the operand size.
 *
 * Note that -P is a project-private interface and together with -G intended
 * to be used only by the growfs script. It is therefore purposely not
 * documented in the man page.
 * The -P option is covered by PSARC case 2003/422.
 */

/*
 * The following constants set the defaults used for the number
 * of sectors/track (fs_nsect), and number of tracks/cyl (fs_ntrak).
 *
 *                      NSECT           NTRAK
 *      72MB CDC        18              9
 *      30MB CDC        18              5
 *      720KB Diskette  9               2
 *
 * However the defaults will be different for disks larger than CHSLIMIT.
 */

#define DFLNSECT        32
#define DFLNTRAK        16

/*
 * The following default sectors and tracks values are used for
 * non-efi disks that are larger than the CHS addressing limit. The
 * existing default cpg of 16 (DESCPG) holds good for larger disks too.
 */
#define DEF_SECTORS_EFI 128
#define DEF_TRACKS_EFI  48

/*
 * The maximum number of cylinders in a group depends upon how much
 * information can be stored on a single cylinder. The default is to
 * use 16 cylinders per group.  This is effectively tradition - it was
 * the largest value acceptable under SunOs 4.1
 */
#define DESCPG          16      /* desired fs_cpg */

/*
 * The following two constants set the default block and fragment sizes.
 * Both constants must be a power of 2 and meet the following constraints:
 *      MINBSIZE <= DESBLKSIZE <= MAXBSIZE
 *      DEV_BSIZE <= DESFRAGSIZE <= DESBLKSIZE
```

```
 128  *      DESBLKSIZE / DESFRAGSIZE <= 8
 129  */
 130 #define DESBLKSIZE      8192
 131 #define DESFRAGSIZE     1024

 133 /*
 134  * MINFREE gives the minimum acceptable percentage of file system
 135  * blocks which may be free. If the freelist drops below this level
 136  * only the superuser may continue to allocate blocks. This may
 137  * be set to 0 if no reserve of free blocks is deemed necessary,
 138  * however throughput drops by fifty percent if the file system
 139  * is run at between 90% and 100% full; thus the default value of
 140  * fs_minfree is 10%. With 10% free space, fragmentation is not a
 141  * problem, so we choose to optimize for time.
 142  */
 143 #define MINFREE         10
 144 #define DEFAULTOPT      FS_OPTTIME

 146 /*
 147  * ROTDELAY gives the minimum number of milliseconds to initiate
 148  * another disk transfer on the same cylinder. It is no longer used
 149  * and will always default to 0.
 150  */
 151 #define ROTDELAY        0

 153 /*
 154  * MAXBLKPG determines the maximum number of data blocks which are
 155  * placed in a single cylinder group. The default is one indirect
 156  * block worth of data blocks.
 157  */
 158 #define MAXBLKPG(bsize) ((bsize) / sizeof (daddr32_t))

 160 /*
 161  * Each file system has a number of inodes statically allocated.
 162  * We allocate one inode slot per NBPI bytes, expecting this
 163  * to be far more than we will ever need.
 164  */
 165 #define NBPI            2048    /* Number Bytes Per Inode */
 166 #define MTB_NBPI        (MB)    /* Number Bytes Per Inode for multi-terabyte */

 168 /*
 169  * Disks are assumed to rotate at 60HZ, unless otherwise specified.
 170  */
 171 #define DEFHZ           60

 173 /*
 174  * Cylinder group related limits.
 175  *
 176  * For each cylinder we keep track of the availability of blocks at different
 177  * rotational positions, so that we can lay out the data to be picked
 178  * up with minimum rotational latency.  NRPOS is the number of rotational
 179  * positions which we distinguish.  With NRPOS 8 the resolution of our
 180  * summary information is 2ms for a typical 3600 rpm drive.
 181  */
 182 #define NRPOS           8       /* number distinct rotational positions */

 184 #ifdef DEBUG
 185 #define dprintf(x)      printf x
 186 #else
 187 #define dprintf(x)
 188 #endif

 190 /*
 191  * For the -N option, when calculating the backup superblocks, do not print
 192  * them if we are not really sure. We may have to try an alternate method of
 193  * arriving at the superblocks. So defer printing till a handful of superblocks
```



```
 194  * look good.
 195  */
 196 #define tprintf(x)      if (Nflag && retry) \
 197                                 (void) strncat(tmpbuf, x, strlen(x)); \
 198                         else \
 199                                 (void) fprintf(stderr, x);

 201 #define ALTSB           32      /* Location of first backup superblock */

 203 /*
 204  * range_check "user_supplied" flag values.
 205  */
 206 #define RC_DEFAULT      0
 207 #define RC_KEYWORD      1
 208 #define RC_POSITIONAL   2

 210 /*
 211  * ufs hole
 212  */
 213 #define UFS_HOLE        -1

 215 #ifndef STANDALONE
 216 #include        <stdio.h>
 217 #include        <sys/mnttab.h>
 218 #endif

 220 #include        <stdlib.h>
 221 #include        <unistd.h>
 222 #include        <malloc.h>
 223 #include        <string.h>
 224 #include        <strings.h>
 225 #include        <ctype.h>
 226 #include        <errno.h>
 227 #include        <sys/param.h>
 228 #include        <time.h>
 229 #include        <sys/types.h>
 230 #include        <sys/sysmacros.h>
 231 #include        <sys/vnode.h>
 232 #include        <sys/fs/ufs_fsdir.h>
 233 #include        <sys/fs/ufs_inode.h>
 234 #include        <sys/fs/ufs_fs.h>
 235 #include        <sys/fs/ufs_log.h>
 236 #include        <sys/mntent.h>
 237 #include        <sys/filio.h>
 238 #include        <limits.h>
 239 #include        <sys/int_const.h>
 240 #include        <signal.h>
 241 #include        <sys/efi_partition.h>
 242 #include        <fslib.h>
 243 #include        "roll_log.h"

 245 #define bcopy(f, t, n)    (void) memcpy(t, f, n)
 246 #define bzero(s, n)     (void) memset(s, 0, n)
 247 #define bcmp(s, d, n)   memcmp(s, d, n)

 249 #define index(s, r)     strchr(s, r)
 250 #define rindex(s, r)    strrchr(s, r)

 252 #include        <sys/stat.h>
 253 #include        <sys/statvfs.h>
 254 #include        <locale.h>
 255 #include        <fcntl.h>
 256 #include        <sys/isa_defs.h>        /* for ENDIAN defines */
 257 #include        <sys/vtoc.h>

 259 #include        <sys/dkio.h>
```

```
260 #include        <sys/asynch.h>

262 extern offset_t llseek();
263 extern char     *getfullblkname();
264 extern long     lrand48();

266 extern int      optind;
267 extern char     *optarg;


270 /*
271  * The size of a cylinder group is calculated by CGSIZE. The maximum size
272  * is limited by the fact that cylinder groups are at most one block.
273  * Its size is derived from the size of the maps maintained in the
274  * cylinder group and the (struct cg) size.
275  */
276 #define CGSIZE(fs) \
277         /* base cg             */ (sizeof (struct cg) + \
278         /* blktot size  */ (fs)->fs_cpg * sizeof (long) + \
279         /* blks size    */ (fs)->fs_cpg * (fs)->fs_nrpos * sizeof (short) + \
280         /* inode map    */ howmany((fs)->fs_ipg, NBBY) + \
281         /* block map */ howmany((fs)->fs_cpg * (fs)->fs_spc / NSPF(fs), NBBY))

283 /*
284  * We limit the size of the inode map to be no more than a
285  * third of the cylinder group space, since we must leave at
286  * least an equal amount of space for the block map.
287  *
288  * N.B.: MAXIpG must be a multiple of INOPB(fs).
289  */
290 #define MAXIpG(fs)      roundup((fs)->fs_bsize * NBBY / 3, INOPB(fs))

292 /*
293  * Same as MAXIpG, but parameterized by the block size (b) and the
294  * cylinder group divisor (d), which is the reciprocal of the fraction of the
295  * cylinder group overhead block that is used for the inode map.  So for
296  * example, if d = 5, the macro's computation assumes that 1/5 of the
297  * cylinder group overhead block can be dedicated to the inode map.
298  */
299 #define MAXIpG_B(b, d)  roundup((b) * NBBY / (d), (b) / sizeof (struct dinode))

301 #define UMASK           0755
302 #define MAXINOPB        (MAXBSIZE / sizeof (struct dinode))
303 #define POWEROF2(num)   (((num) & ((num) - 1)) == 0)
304 #define MB              (1024*1024)
305 #define BETWEEN(x, l, h)        ((x) >= (l) && (x) <= (h))

307 /*
308  * Used to set the inode generation number. Since both inodes and dinodes
309  * are dealt with, we really need a pointer to an icommon here.
310  */
311 #define IRANDOMIZE(icp) (icp)->ic_gen = lrand48();

313 /*
314  * Flags for number()
315  */
316 #define ALLOW_PERCENT   0x01    /* allow trailing '%' on number */
317 #define ALLOW_MS1       0x02    /* allow trailing 'ms', state 1 */
318 #define ALLOW_MS2       0x04    /* allow trailing 'ms', state 2 */
319 #define ALLOW_END_ONLY  0x08    /* must be at end of number & suffixes */

321 #define MAXAIO  1000    /* maximum number of outstanding I/O's we'll manage */
322 #define BLOCK   1       /* block in aiowait */
323 #define NOBLOCK 0       /* don't block in aiowait */

325 #define RELEASE 1       /* free an aio buffer after use */
```



```
326 #define SAVE    0       /* don't free the buffer */

328 typedef struct aio_trans {
329         aio_result_t resultbuf;
330         diskaddr_t bno;
331         char *buffer;
332         int size;
333         int release;
334         struct aio_trans *next;
335 } aio_trans;
```

unchanged_portion_omitted_