```
**********************************************************
    5803 Mon Jan 21 16:34:39 2019
new/usr/src/cmd/auditstat/auditstat.c
10121 smatch fix for auditstat
**********************************************************
/*
 * 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 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

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

#include "statcommon.h"

#include <sys/types.h>
#include <stdlib.h>
#include <ctype.h>
#include <stdio.h>
#include <bsm/audit.h>
#include <bsm/libbsm.h>
#include <unistd.h>
#include <locale.h>

#if !defined(TEXT_DOMAIN)               /* Should be defined by cc -D */
#define TEXT_DOMAIN "SYS_TEST"          /* Use this only if it isn’t */
#endif


/*
 * Display header every HEADER_MOD lines printed
 */
#define         DFLT_HEADER_MOD (20)
#define         ONEK (1024)

#define         CFLG (0x01)
#define         HFLG (0x02)
#define         IFLG (0x04)
#define         NFLG (0x08)
#define         VFLG (0x10)

extern char     *optarg;

static int      count;
static int      flags;
```

```
static int      header_mod = DFLT_HEADER_MOD;
static int      interval;
static uint_t timestamp_fmt = NODATE;

static void     display_stats();
static void     eauditon();
static void     parse_args();
static void     usage_exit();
static int      strisdigit();

int
main(argc, argv)
int     argc;
char    **argv;
{
        register int    i;
        au_stat_t s;

        (void) setlocale(LC_ALL, "");
        (void) textdomain(TEXT_DOMAIN);

        (void) setbuf(stdout, (char *)0);
        (void) setbuf(stderr, (char *)0);

        parse_args(argc, argv);

        if (!flags) {
                eauditon(A_GETSTAT, (caddr_t)&s, NULL);
                if (timestamp_fmt != NODATE)
                        print_timestamp(timestamp_fmt);
                display_stats(&s, 0);
                exit(0);
        }

        if (flags & VFLG || flags & NFLG)
                eauditon(A_GETSTAT, (caddr_t)&s, NULL);

        if (flags & VFLG)
                (void) printf("version = %d\n", s.as_version);

        if (flags & NFLG)
                (void) printf("number of kernel events = %d\n", s.as_numevent);

        if (!(flags & IFLG))
                exit(0);

        /* CSTYLED */
        for (i = 0;; i++) {
                eauditon(A_GETSTAT, (caddr_t)&s, NULL);
                if (timestamp_fmt != NODATE)
                        print_timestamp(timestamp_fmt);
                display_stats(&s, i);
                if ((flags & CFLG) && count)
                        if (i == count - 1)
                                break;
                (void) sleep(interval);
        }

        return (0);
}


static void
display_stats(au_stat_t *s, int cnt)
display_stats(s, cnt)
au_stat_t *s;
```

```
{
        int     offset[12];   /* used to line the header up correctly */
        char    buf[512];

        (void) sprintf(buf,
"%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u %n%4u%n",
                s->as_generated,        &(offset[0]),
                s->as_nonattrib,        &(offset[1]),
                s->as_kernel,           &(offset[2]),
                s->as_audit,            &(offset[3]),
                s->as_auditctl,         &(offset[4]),
                s->as_enqueue,          &(offset[5]),
                s->as_written,          &(offset[6]),
                s->as_wblocked,         &(offset[7]),
                s->as_rblocked,         &(offset[8]),
                s->as_dropped,          &(offset[9]),
                s->as_totalsize / ONEK, &(offset[10]),
                s->as_memused / ONEK,   &(offset[11]));

        /* print a properly aligned header every HEADER_MOD lines */
        if (header_mod && (!cnt || ((timestamp_fmt != NODATE) ?
            !(cnt % (header_mod / 2)) : !(cnt % header_mod)))) {
                (void) printf(
                        "%*s %*s %*s %*s %*s %*s %*s %*s %*s %*s %*s %*s\n",
                        offset[0] - 1,                  "gen",
                        offset[1] - offset[0] - 1,      "nona",
                        offset[2] - offset[1] - 1,      "kern",
                        offset[3] - offset[2] - 1,      "aud",
                        offset[4] - offset[3] - 1,      "ctl",
                        offset[5] - offset[4] - 1,      "enq",
                        offset[6] - offset[5] - 1,      "wrtn",
                        offset[7] - offset[6] - 1,      "wblk",
                        offset[8] - offset[7] - 1,      "rblk",
                        offset[9] - offset[8] - 1,      "drop",
                        offset[10] - offset[9] - 1,     "tot",
                        offset[11] - offset[10],        "mem");
        }

        (void) puts(buf);
}
```

______***unchanged_portion_omitted_***