```
**********************************************************
   20574 Wed Apr 29 08:42:15 2015
new/usr/src/lib/libc/port/gen/privlib.c
5763 missing va_end() on return from __fini_daemon_priv()
**********************************************************
```

```
/*
 * 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 2015 Gary Mills
 * Copyright (c) 2003, 2010, Oracle and/or its affiliates. All rights reserved.
 */

#pragma weak _getprivimplinfo   = getprivimplinfo
#pragma weak _priv_addset       = priv_addset
#pragma weak _priv_allocset     = priv_allocset
#pragma weak _priv_copyset      = priv_copyset
#pragma weak _priv_delset       = priv_delset
#pragma weak _priv_emptyset     = priv_emptyset
#pragma weak _priv_basicset     = priv_basicset
#pragma weak _priv_fillset      = priv_fillset
#pragma weak _priv_freeset      = priv_freeset
#pragma weak _priv_getbyname    = priv_getbyname
#pragma weak _priv_getbynum     = priv_getbynum
#pragma weak _priv_getsetbyname = priv_getsetbyname
#pragma weak _priv_getsetbynum  = priv_getsetbynum
#pragma weak _priv_ineffect     = priv_ineffect
#pragma weak _priv_intersect    = priv_intersect
#pragma weak _priv_inverse      = priv_inverse
#pragma weak _priv_isemptyset   = priv_isemptyset
#pragma weak _priv_isequalset   = priv_isequalset
#pragma weak _priv_isfullset    = priv_isfullset
#pragma weak _priv_ismember     = priv_ismember
#pragma weak _priv_issubset     = priv_issubset
#pragma weak _priv_set          = priv_set
#pragma weak _priv_union        = priv_union

#include "lint.h"

#define _STRUCTURED_PROC        1

#include "priv_private.h"
#include "mtlib.h"
#include "libc.h"
#include <errno.h>
#include <stdarg.h>
#include <stdlib.h>
#include <unistd.h>
```



```
#include <strings.h>
#include <synch.h>
#include <alloca.h>
#include <atomic.h>
#include <sys/ucred.h>
#include <sys/procfs.h>
#include <sys/param.h>
#include <sys/corectl.h>
#include <priv_utils.h>
#include <zone.h>

/* Include each string only once - until the compiler/linker are fixed */
static const char *permitted    = PRIV_PERMITTED;
static const char *effective    = PRIV_EFFECTIVE;
static const char *limit        = PRIV_LIMIT;
static const char *inheritable  = PRIV_INHERITABLE;
/*
 * Data independent privilege set operations.
 *
 * Only a few functions are provided that do not default to
 * the system implementation of privileges.  A limited set of
 * interfaces is provided that accepts a priv_data_t *
 * argument; this set of interfaces is a private interface between libc
 * and libproc.  It is delivered in order to interpret privilege sets
 * in debuggers in a implementation independent way.  As such, we
 * don't need to provide the bulk of the interfaces, only a few
 * boolean tests (isfull, isempty) the name<->num mappings and
 * set pretty print functions.   The boolean tests are only needed for
 * the latter, so those aren't provided externally.
 *
 * Additionally, we provide the function that maps the kernel implementation
 * structure into a libc private data structure.
 */

priv_data_t *privdata;

static mutex_t pd_lock = DEFAULTMUTEX;

static int
parseninfo(priv_info_names_t *na, char ***buf, int *cp)
{
        char *q;
        int i;

        *buf = libc_malloc(sizeof (char *) * na->cnt);

        if (*buf == NULL)
                return (-1);

        q = na->names;

        for (i = 0; i < na->cnt; i++) {
                int l = strlen(q);

                (*buf)[i] = q;
                q += l + 1;
        }
        *cp = na->cnt;
        return (0);
}
______unchanged_portion_omitted_

/*
 * The routine __fini_daemon_priv() is private to Solaris and is
 * used by daemons to clear remaining unwanted privileges and
 * reenable core dumps.
```

```
 */
void
__fini_daemon_priv(const char *priv, ...)
{
        priv_set_t *nset;
        va_list pa;

        if (priv != NULL) {

                va_start(pa, priv);
                nset = priv_vlist(pa);
                va_end(pa);

        if (priv != NULL) {
                nset = priv_vlist(pa);
                if (nset == NULL)
                        return;

                (void) priv_addset(nset, priv);
                (void) setppriv(PRIV_OFF, permitted, nset);
                priv_freeset(nset);
        }

        va_end(pa);

        (void) setpflags(__PROC_PROTECT, 0);
}
```

_____unchanged_portion_omitted_