```
**********************************************************
   79209 Sat Oct 19 13:05:44 2013
new/usr/src/cmd/mdb/common/mdb/mdb_cmds.c
patch mdb_var_alloc
**********************************************************
______unchanged_portion_omitted_

1546 /*ARGSUSED*/
1547 static int
1548 showrev_addversion(void *vers_nv, const mdb_map_t *ignored, const char *object)
1549 {
1550         ctf_file_t *ctfp;
1551         const char *version = NULL;
1552         char *objname;

1554         objname = mdb_alloc(strlen(object) + 1, UM_SLEEP | UM_GC);
1555         (void) strcpy(objname, object);

1557         if ((ctfp = mdb_tgt_name_to_ctf(mdb.m_target, objname)) != NULL)
1558                 version = ctf_label_topmost(ctfp);

1560         /*
1561          * Not all objects have CTF and label data, so set version to "Unknown".
1562          */
1563         if (version == NULL)
1564                 version = "Unknown";

1566         /*
1567          * The hash table implementation in OVERLOAD mode limits the version
1568          * name to 31 characters because we cannot specify an external name.
1569          * The full version name is available via the ::objects dcmd if needed.
1570          */
1566         (void) mdb_nv_insert(vers_nv, version, NULL, (uintptr_t)objname,
1567             MDB_NV_OVERLOAD);

1569         return (0);
1570 }
______unchanged_portion_omitted_
```

```
**********************************************************
   10090 Sat Oct 19 13:05:45 2013
new/usr/src/cmd/mdb/common/mdb/mdb_nv.c
patch mdb_var_alloc
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License, Version 1.0 only
 * (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 2004 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

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

#include <mdb/mdb_debug.h>
#include <mdb/mdb_string.h>
#include <mdb/mdb_modapi.h>
#include <mdb/mdb_err.h>
#include <mdb/mdb_nv.h>
#include <mdb/mdb.h>

#define NV_NAME(v) \
        (((v)->v_flags & MDB_NV_EXTNAME) ? (v)->v_ename : (v)->v_lname)

#define NV_SIZE(v) \
        (((v)->v_flags & MDB_NV_EXTNAME) ? sizeof (mdb_var_t) : \
        sizeof (mdb_var_t) + strlen((v)->v_lname))
        sizeof (mdb_var_t) + MDB_NV_NAMELEN - 1)

#define NV_HASHSZ       211

static size_t
nv_hashstring(const char *key)
{
        size_t g, h = 0;
        const char *p;

        ASSERT(key != NULL);

        for (p = key; *p != '\0'; p++) {
                h = (h << 4) + *p;

                if ((g = (h & 0xf0000000)) != 0) {
                        h ^= (g >> 24);
                        h ^= g;
                }
        }
```



```
        return (h);
}

static mdb_var_t *
nv_var_alloc(const char *name, const mdb_nv_disc_t *disc,
        uintmax_t value, uint_t flags, uint_t um_flags, mdb_var_t *next)
{
        size_t nbytes;
        mdb_var_t *v;

        if (flags & MDB_NV_EXTNAME)
                nbytes = sizeof (mdb_var_t);
        else
                nbytes = sizeof (mdb_var_t) + strlen(name);
        size_t nbytes = (flags & MDB_NV_EXTNAME) ? sizeof (mdb_var_t) :
            (sizeof (mdb_var_t) + MDB_NV_NAMELEN - 1);

        v = mdb_alloc(nbytes, um_flags);
        mdb_var_t *v = mdb_alloc(nbytes, um_flags);

        if (v == NULL)
                return (NULL);

        if (flags & MDB_NV_EXTNAME) {
                v->v_ename = name;
                v->v_lname[0] = '\0';
                v->v_lname[0] = 0;
        } else {
                strcpy(v->v_lname, name);
                (void) strncpy(v->v_lname, name, MDB_NV_NAMELEN - 1);
                v->v_lname[MDB_NV_NAMELEN - 1] = '\0';
                v->v_ename = NULL;
        }

        v->v_uvalue = value;
        v->v_flags = flags & ~(MDB_NV_SILENT | MDB_NV_INTERPOS);
        v->v_disc = disc;
        v->v_next = next;

        return (v);
}
```

_____*unchanged_portion_omitted_*

```
**********************************************************
   14818 Sat Oct 19 13:05:45 2013
new/usr/src/cmd/mdb/common/mdb/mdb_tab.c
patch mdb_var_alloc
**********************************************************
______unchanged_portion_omitted_

 389 /*
 390  * Determine whether the specified name is a valid tab completion for
 391  * the given command. If the name is a valid tab completion then
 392  * it will be saved in the mdb_tab_cookie_t.
 393  */
 394 void
 395 mdb_tab_insert(mdb_tab_cookie_t *mcp, const char *name)
 396 {
 397         size_t matches, index;
 397         size_t len, matches, index;
 398         uint_t flags;
 398         mdb_var_t *v;
 399         char *n;
 401         const char *nvn;

 401         /*
 402          * If we have a match set, then we want to verify that we actually match
 403          * it.
 404          */
 405         if (mcp->mtc_base != NULL &&
 406             strncmp(name, mcp->mtc_base, strlen(mcp->mtc_base)) != 0)
 407                 return;

 409         v = mdb_nv_lookup(&mcp->mtc_nv, name);
 410         if (v != NULL)
 411                 return;

 413         (void) mdb_nv_insert(&mcp->mtc_nv, name, NULL, 0, MDB_NV_RDONLY);
 415         /*
 416          * Names that we get passed in may be longer than MDB_NV_NAMELEN which
 417          * is currently 31 including the null terminator. If that is the case,
 418          * then we're going to take care of allocating a string and holding it
 419          * for our caller. Note that we don't need to free it, because we're
 420          * allocating this with UM_GC.
 421          */
 422         flags = 0;
 423         len = strlen(name);
 424         if (len > MDB_NV_NAMELEN - 1) {
 425                 n = mdb_alloc(len + 1, UM_SLEEP | UM_GC);
 426                 (void) strcpy(n, name);
 427                 nvn = n;
 428                 flags |= MDB_NV_EXTNAME;
 429         } else {
 430                 nvn = name;
 431         }
 432         flags |= MDB_NV_RDONLY;

 434         (void) mdb_nv_insert(&mcp->mtc_nv, nvn, NULL, 0, flags);

 415         matches = mdb_tab_size(mcp);
 416         if (matches == 1) {
 417                 (void) strlcpy(mcp->mtc_match, name, MDB_SYM_NAMLEN);
 438                 (void) strlcpy(mcp->mtc_match, nvn, MDB_SYM_NAMLEN);
 418         } else {
 419                 index = 0;
 420                 while (mcp->mtc_match[index] &&
 421                     mcp->mtc_match[index] == name[index])
 442                     mcp->mtc_match[index] == nvn[index])
 422                         index++;
```



```
 424                 mcp->mtc_match[index] = '\0';
 425         }
 426 }
______unchanged_portion_omitted_
```