```
**********************************************************
   11109 Wed May  8 02:34:15 2019
new/usr/src/uts/common/os/subr.c
10907 hot_patch_kernel_text() has no respect for boundaries
Reviewed by: Jerry Jelinek <jerry.jelinek@joyent.com>
Reviewed by: Patrick Mooney <patrick.mooney@joyent.com>
Reviewed by: Robert Mustacchi <rm@joyent.com>
**********************************************************
```

```
/*
 * 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 2019 Joyent, Inc.
 */

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

#include <sys/types.h>
#include <sys/sysmacros.h>
#include <sys/param.h>
#include <sys/vmparam.h>
#include <sys/systm.h>
#include <sys/cred.h>
#include <sys/user.h>
#include <sys/proc.h>
#include <sys/conf.h>
#include <sys/tuneable.h>
#include <sys/cpuvar.h>
#include <sys/archsystm.h>
#include <sys/vmem.h>
#include <vm/seg_kmem.h>
#include <sys/errno.h>
#include <sys/cmn_err.h>
#include <sys/debug.h>
#include <sys/atomic.h>
#include <sys/model.h>
#include <sys/kmem.h>
#include <sys/memlist.h>
#include <sys/autoconf.h>
#include <sys/ontrap.h>
#include <sys/utsname.h>
#include <sys/zone.h>
```



```
#ifdef __sparc
#include <sys/membar.h>
#endif

/*
 * Routine which sets a user error; placed in
 * illegal entries in the bdevsw and cdevsw tables.
 */

int
nodev()
{
        return (curthread->t_lwp ?
            ttolwp(curthread)->lwp_error = ENXIO : ENXIO);
}
```

*unchanged_portion_omitted_*

```
/*
 * Hot-patch a single instruction in the kernel's text.
 *
 * If you want to patch multiple instructions you must arrange to do it so that
 * all intermediate stages are sane -- we don't stop other cpus while doing
 * this.
 *
 * If you want to patch multiple instructions you must
 * arrange to do it so that all intermediate stages are
 * sane -- we don't stop other cpus while doing this.
 * Size must be 1, 2, or 4 bytes with iaddr aligned accordingly.
 *
 * The instruction itself might straddle a page boundary, so we have to account
 * for that.
 */
void
hot_patch_kernel_text(caddr_t iaddr, uint32_t new_instr, uint_t size)
{
        const uintptr_t pageoff = (uintptr_t)iaddr & PAGEOFFSET;
        const boolean_t straddles = (pageoff + size > PAGESIZE);
        const size_t mapsize = straddles ? PAGESIZE * 2 : PAGESIZE;
        caddr_t ipageaddr = iaddr - pageoff;
        caddr_t vaddr;
        page_t **ppp;
        uintptr_t off = (uintptr_t)iaddr & PAGEOFFSET;

        vaddr = vmem_alloc(heap_arena, mapsize, VM_SLEEP);
        vaddr = vmem_alloc(heap_arena, PAGESIZE, VM_SLEEP);

        (void) as_pagelock(&kas, &ppp, ipageaddr, mapsize, S_WRITE);
        (void) as_pagelock(&kas, &ppp, iaddr - off, PAGESIZE, S_WRITE);

        hat_devload(kas.a_hat, vaddr, PAGESIZE,
            hat_getpfnum(kas.a_hat, ipageaddr), PROT_READ | PROT_WRITE,
            HAT_LOAD_LOCK | HAT_LOAD_NOCONSIST);

        if (straddles) {
                hat_devload(kas.a_hat, vaddr + PAGESIZE, PAGESIZE,
                    hat_getpfnum(kas.a_hat, ipageaddr + PAGESIZE),
                hat_getpfnum(kas.a_hat, iaddr - off),
                        PROT_READ | PROT_WRITE, HAT_LOAD_LOCK | HAT_LOAD_NOCONSIST);
        }

        switch (size) {
        case 1:
                *(uint8_t *)(vaddr + pageoff) = new_instr;
                *(uint8_t *)(vaddr + off) = new_instr;
                break;
        case 2:
```

```
354                 *(uint16_t *)(vaddr + pageoff) = new_instr;
336                 *(uint16_t *)(vaddr + off) = new_instr;
355                 break;
356         case 4:
357                 *(uint32_t *)(vaddr + pageoff) = new_instr;
339                 *(uint32_t *)(vaddr + off) = new_instr;
358                 break;
359         default:
360                 panic("illegal hot-patch");
361         }

363         membar_enter();
364         sync_icache(vaddr + pageoff, size);
346         sync_icache(vaddr + off, size);
365         sync_icache(iaddr, size);
366         as_pageunlock(&kas, ppp, ipageaddr, mapsize, S_WRITE);
367         hat_unload(kas.a_hat, vaddr, mapsize, HAT_UNLOAD_UNLOCK);
368         vmem_free(heap_arena, vaddr, mapsize);
348         as_pageunlock(&kas, ppp, iaddr - off, PAGESIZE, S_WRITE);
349         hat_unload(kas.a_hat, vaddr, PAGESIZE, HAT_UNLOAD_UNLOCK);
350         vmem_free(heap_arena, vaddr, PAGESIZE);
369 }
```

_____unchanged_portion_omitted_