****
    7179 Thu Oct 11 22:38:17 2012
new/usr/src/cmd/sgs/elfdump/common/_elfdump.h
3265 link-editor builds bogus .eh_frame_hdr on ia32
****

______unchanged_portion_omitted_

```
/*
 * Define various elfdump() functions into their 32-bit and 64-bit variants.
 */
#if     defined(_ELF64)
#define cap                     cap64
#define checksum                checksum64
#define dynamic                 dynamic64
#define fake_shdr_cache         fake_shdr_cache64
#define fake_shdr_cache_free    fake_shdr_cache_free64
#define got                     got64
#define group                   group64
#define hash                    hash64
#define interp                  interp64
#define move                    move64
#define note                    note64
#define note_entry              note_entry64
#define regular                 regular64
#define reloc                   reloc64
#define sections                sections64
#define string                  string64
#define symbols                 symbols64
#define syminfo                 syminfo64
#define symlookup               symlookup64
#define unwind                  unwind64
#define versions                versions64
#define version_def             version_def64
#define version_need            version_need64
#else
#define cap                     cap32
#define checksum                checksum32
#define dynamic                 dynamic32
#define fake_shdr_cache         fake_shdr_cache32
#define fake_shdr_cache_free    fake_shdr_cache_free32
#define got                     got32
#define group                   group32
#define hash                    hash32
#define interp                  interp32
#define move                    move32
#define note                    note32
#define note_entry              note_entry32
#define regular                 regular32
#define reloc                   reloc32
#define sections                sections32
#define string                  string32
#define symbols                 symbols32
#define syminfo                 syminfo32
#define symlookup               symlookup32
#define unwind                  unwind32
#define versions                versions32
#define version_def             version_def32
#define version_need            version_need32
#endif

extern  corenote_ret_t  corenote(Half, int, Word, const char *, Word);
extern  void    dump_eh_frame(uchar_t *, size_t, uint64_t, Half e_machine,
                    uchar_t *e_ident, uint64_t gotaddr);
                    uchar_t *e_ident);
extern  void    dump_hex_bytes(const void *, size_t, int, int, int);
```

```
extern  int     fake_shdr_cache32(const char *, int, Elf *, Elf32_Ehdr *,
                    Cache **, size_t *);
extern  int     fake_shdr_cache64(const char *, int, Elf *, Elf64_Ehdr *,
                    Cache **, size_t *);

extern  void    fake_shdr_cache_free32(Cache *, size_t);
extern  void    fake_shdr_cache_free64(Cache *, size_t);

extern  int     regular32(const char *, int, Elf *, uint_t, const char *, int,
                    uchar_t);
extern  int     regular64(const char *, int, Elf *, uint_t, const char *, int,
                    uchar_t);

#ifdef  __cplusplus
}
```

______unchanged_portion_omitted_

```
***********************************************************
   19716 Thu Oct 11 22:38:17 2012
new/usr/src/cmd/sgs/elfdump/common/dwarf.c
3265 link-editor builds bogus .eh_frame_hdr on ia32
***********************************************************
/*
 * 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.
 */

#include        <_libelf.h>
#include        <dwarf.h>
#include        <stdio.h>
#include        <unistd.h>
#include        <errno.h>
#include        <strings.h>
#include        <debug.h>
#include        <conv.h>
#include        <msg.h>
#include        <_elfdump.h>


/*
 * Data from eh_frame section used by dump_cfi()
 */
typedef struct {
        Half            e_machine;      /* ehdr->e_machine */
        uchar_t         *e_ident;       /* ehdr->e_ident */
        uint64_t        sh_addr;        /* Address of eh_frame section */
        int             do_swap;        /* True if object and system byte */
                                        /*      order differs */
        int             cieRflag;       /* R flag from current CIE */
        uint64_t        ciecalign;      /* CIE code align factor */
        int64_t         ciedalign;      /* CIE data align factor */
        uint64_t        fdeinitloc;     /* FDE initial location */
        uint64_t        gotaddr;        /* Address of the GOT */
#endif /* ! codereview */
} dump_cfi_state_t;


/*
 * Extract an unsigned integer value from an .eh_frame section, converting it
 * from its native byte order to that of the running machine if necessary.
 *
 * entry:
```

```
 *      data - Base address from which to extract datum
 *      ndx - Address of variable giving index to start byte in data.
 *      size - # of bytes in datum. Must be one of: 1, 2, 4, 8
 *      do_swap - True if the data is in a different byte order than that
 *              of the host system.
 *
 * exit:
 *      *ndx is incremented by the size of the extracted datum.
 *
 *      The requested datum is extracted, byte swapped if necessary,
 *      and returned.
 */
static uint64_t
dwarf_extract_uint(uchar_t *data, uint64_t *ndx, int size, int do_swap)
{
        switch (size) {
        case 1:
                return (data[(*ndx)++]);
        case 2:
                {
                        Half    r;
                        uchar_t *p = (uchar_t *)&r;

                        data += *ndx;
                        if (do_swap)
                                UL_ASSIGN_BSWAP_HALF(p, data);
                        else
                                UL_ASSIGN_HALF(p, data);

                        (*ndx) += 2;
                        return (r);
                }
        case 4:
                {
                        Word    r;
                        uchar_t *p = (uchar_t *)&r;

                        data += *ndx;
                        if (do_swap)
                                UL_ASSIGN_BSWAP_WORD(p, data);
                        else
                                UL_ASSIGN_WORD(p, data);

                        (*ndx) += 4;
                        return (r);
                }

        case 8:
                {
                        uint64_t        r;
                        uchar_t         *p = (uchar_t *)&r;

                        data += *ndx;
                        if (do_swap)
                                UL_ASSIGN_BSWAP_LWORD(p, data);
                        else
                                UL_ASSIGN_LWORD(p, data);

                        (*ndx) += 8;
                        return (r);
                }
        }

        /* If here, an invalid size was specified */
        assert(0);
        return (0);
```

```c
}

/*
 * Map a DWARF register constant to the machine register name it
 * corresponds to, formatting the result into buf.
 *
 * The assignment of DWARF register numbers is part of the system
 * specific ABI for each platform.
 *
 * entry:
 *      regno - DWARF register number
 *      mach - ELF machine code for platform
 *      buf, bufsize - Buffer to receive the formatted result string
 *
 * exit:
 *      The results are formatted into buf, and buf is returned.
 *      If the generated output would exceed the size of the buffer
 *      provided, it will be clipped to fit.
 */
static const char *
dwarf_regname(Half mach, int regno, char *buf, size_t bufsize)
{
        Conv_inv_buf_t  inv_buf;
        const char      *name;
        int             good_name;

        name = conv_dwarf_regname(mach, regno, 0, &good_name, &inv_buf);

        /*
         * If there is a good mnemonic machine name for the register,
         * format the result as 'r# (mnemonic)'.  If there is no good
         * name for it, then simply format the dwarf name as 'r#'.
         */
        if (good_name)
                (void) snprintf(buf, bufsize, MSG_ORIG(MSG_REG_FMT_NAME),
                    regno, name);
        else
                (void) snprintf(buf, bufsize, MSG_ORIG(MSG_REG_FMT_BASIC),
                    regno);

        return (buf);
}


/*
 * Decode eh_frame Call Frame Instructions, printing each one on a
 * separate line.
 *
 * entry:
 *      data - Address of base of eh_frame section being processed
 *      off - Offset of current FDE within eh_frame
 *      ndx - Index of current position within current FDE
 *      len - Length of eh_frame section
 *      state - Object, CIE, and FDE state for current request
 *      msg - Header message to issue before producing output.
 *      indent - # of indentation characters issued for each line of output.
 *
 * exit:
 *      The Call Frame Instructions have been decoded and printed.
 *
 *      *ndx has been incremented to contain the index of the next
 *              byte of data to be processed in eh_frame.
 *
 * note:
 *      The format of Call Frame Instructions in .eh_frame sections is based
 *      on the DWARF specification.
```

```c
 */
static void
dump_cfi(uchar_t *data, uint64_t off, uint64_t *ndx, uint_t len,
    dump_cfi_state_t *state, const char *msg, int indent)
{
        /*
         * We use %*s%s to insert leading whitespace and the op name.
         * PREFIX supplies these arguments.
         */
#define PREFIX  indent, MSG_ORIG(MSG_STR_EMPTY), opname

        /* Hide boilerplate clutter in calls to dwarf_regname() */
#define REGNAME(_rnum, _buf) \
        dwarf_regname(state->e_machine, _rnum, _buf, sizeof (_buf))

        /* Extract the lower 6 bits from an op code */
#define LOW_OP(_op) (_op & 0x3f)

        char            rbuf1[32], rbuf2[32];
        Conv_inv_buf_t  inv_buf;
        uchar_t         op;
        const char      *opname;
        uint64_t        oper1, oper2, cur_pc;
        int64_t         soper;
        const char      *loc_str;
        int             i;

        dbg_print(0, msg);

        /*
         * In a CIE/FDE, the length field does not include it's own
         * size. Hence, the value passed in is 4 less than the index
         * of the actual final location.
         */
        len += 4;

        /*
         * There is a concept of the 'current location', which is the PC
         * to which the current item applies. It starts out set to the
         * FDE initial location, and can be set or incremented by
         * various OP codes. cur_pc is used to track this.
         *
         * We want to use 'initloc' in the output the first time the location
         * is referenced, and then switch to 'loc' for subsequent references.
         * loc_str is used to manage that.
         */
        cur_pc = state->fdeinitloc;
        loc_str = MSG_ORIG(MSG_STR_INITLOC);

        while (*ndx < len) {
                /*
                 * The first byte contains the primary op code in the top
                 * 2 bits, so there are 4 of them. Primary OP code
                 * 0 uses the lower 6 bits to specify a sub-opcode, allowing
                 * for 64 of them. The other 3 primary op codes use the
                 * lower 6 bits to hold an operand (a register #, or value).
                 *
                 * Check the primary OP code. If it's 1-3, handle it
                 * and move to the next loop iteration. For OP code 0,
                 * fall through to decode the sub-code.
                 */
                op = data[off + (*ndx)++];
                opname = conv_dwarf_cfa(op, 0, &inv_buf);
                switch (op >> 6) {
                case 0x1:               /* v2: DW_CFA_advance_loc, delta */
                        oper1 = state->ciecalign * LOW_OP(op);
                        cur_pc += oper1;
                        dbg_print(0, MSG_ORIG(MSG_CFA_ADV_LOC), PREFIX,
                            loc_str, EC_XWORD(oper1), EC_XWORD(cur_pc));
                        loc_str = MSG_ORIG(MSG_STR_LOC);
                        continue;

                case 0x2:               /* v2: DW_CFA_offset, reg, offset */
                        soper = uleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_CFAOFF), PREFIX,
                            REGNAME(LOW_OP(op), rbuf1), EC_SXWORD(soper));
                        continue;

                case 0x3:               /* v2: DW_CFA_restore, reg */
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG), PREFIX,
                            REGNAME(LOW_OP(op), rbuf1));
                        continue;
                }

                /*
                 * If we’re here, the high order 2 bits are 0. The low 6 bits
                 * specify a sub-opcode defining the operation.
                 */
                switch (op) {
                case 0x00:              /* v2: DW_CFA_nop */
                        /*
                         * No-ops are used to fill unused space required
                         * for alignment. It is common for there to be
                         * multiple adjacent nops. It saves space to report
                         * them all with a single line of output.
                         */
                        for (i = 1;
                            (*ndx < len) && (data[off + *ndx] == 0);
                            i++, (*ndx)++)
                                ;
                        dbg_print(0, MSG_ORIG(MSG_CFA_SIMPLEREP), PREFIX, i);
                        break;

                case 0x0a:              /* v2: DW_CFA_remember_state */
                case 0x0b:              /* v2: DW_CFA_restore_state */
                case 0x2d:              /* GNU: DW_CFA_GNU_window_save */
                        dbg_print(0, MSG_ORIG(MSG_CFA_SIMPLE), PREFIX);
                        break;

                case 0x01:              /* v2: DW_CFA_set_loc, address */
                        cur_pc = dwarf_ehe_extract(&data[off], ndx,
                            state->cieRflag, state->e_ident, B_FALSE,
                            state->sh_addr, off + *ndx, state->gotaddr);
                            state->cieRflag, state->e_ident,
                            state->sh_addr, off + *ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_CFASET), PREFIX,
                            EC_XWORD(cur_pc));
                        break;

                case 0x02:      /* v2: DW_CFA_advance_loc_1, 1-byte delta */
                case 0x03:      /* v2: DW_CFA_advance_loc_2, 2-byte delta */
                case 0x04:      /* v2: DW_CFA_advance_loc_4, 4-byte delta */
                        /*
                         * Since the codes are contiguous, and the sizes are
                         * powers of 2, we can compute the word width from
                         * the code.
                         */
                        i = 1 << (op - 0x02);
                        oper1 = dwarf_extract_uint(data + off, ndx, i,
                            state->do_swap) * state->ciecalign;
                        cur_pc += oper1;
```

```
                        dbg_print(0, MSG_ORIG(MSG_CFA_ADV_LOC), PREFIX,
                            loc_str, EC_XWORD(oper1), EC_XWORD(cur_pc));
                        loc_str = MSG_ORIG(MSG_STR_LOC);
                        break;

                case 0x05:              /* v2: DW_CFA_offset_extended,reg,off */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = uleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_CFAOFF), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                case 0x06:              /* v2: DW_CFA_restore_extended, reg */
                case 0x0d:              /* v2: DW_CFA_def_cfa_register, reg */
                case 0x08:              /* v2: DW_CFA_same_value, reg */
                case 0x07:              /* v2: DW_CFA_undefined, reg */
                        oper1 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG), PREFIX,
                            REGNAME(oper1, rbuf1));
                        break;


                case 0x09:              /* v2: DW_CFA_register, reg, reg */
                        oper1 = uleb_extract(&data[off], ndx);
                        oper2 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_REG), PREFIX,
                            REGNAME(oper1, rbuf1), REGNAME(oper2, rbuf2));
                        break;

                case 0x0c:              /* v2: DW_CFA_def_cfa, reg, offset */
                        oper1 = uleb_extract(&data[off], ndx);
                        oper2 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_OFFLLU), PREFIX,
                            REGNAME(oper1, rbuf1), EC_XWORD(oper2));
                        break;

                case 0x0e:              /* v2: DW_CFA_def_cfa_offset, offset */
                        oper1 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_LLU), PREFIX,
                            EC_XWORD(oper1));
                        break;

                case 0x0f:              /* v3: DW_CFA_def_cfa_expression, blk */
                        oper1 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_EBLK), PREFIX,
                            EC_XWORD(oper1));
                        /* We currently do not decode the expression block */
                        *ndx += oper1;
                        break;

                case 0x10:              /* v3: DW_CFA_expression, reg, blk */
                case 0x16:              /* v3: DW_CFA_val_expression,reg,blk */
                        oper1 = uleb_extract(&data[off], ndx);
                        oper2 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_EBLK), PREFIX,
                            REGNAME(oper1, rbuf1), EC_XWORD(oper2));
                        /* We currently do not decode the expression block */
                        *ndx += oper2;
                        break;

                case 0x11:      /* v3: DW_CFA_offset_extended_sf, reg, off */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = sleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_CFAOFF), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                case 0x12:              /* v3: DW_CFA_def_cfa_sf, reg, offset */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = sleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_OFFLLD), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                case 0x13:              /* DW_CFA_def_cfa_offset_sf, offset */
                        soper = sleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_LLD), PREFIX,
                            EC_SXWORD(soper));
                        break;

                case 0x14:              /* v3: DW_CFA_val_offset, reg, offset */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = uleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_OFFLLD), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                case 0x15:      /* v3: DW_CFA_val_offset_sf, reg, offset */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = sleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_REG_OFFLLD), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                case 0x1d:      /* GNU: DW_CFA_MIPS_advance_loc8, delta */
                        oper1 = dwarf_extract_uint(data + off, ndx, i,
                            state->do_swap) * state->ciecalign;
                        cur_pc += oper1;
                        dbg_print(0, MSG_ORIG(MSG_CFA_ADV_LOC), PREFIX,
                            loc_str, EC_XWORD(oper1), EC_XWORD(cur_pc));
                        loc_str = MSG_ORIG(MSG_STR_LOC);
                        break;

                case 0x2e:              /* GNU: DW_CFA_GNU_args_size, size */
                        oper1 = uleb_extract(&data[off], ndx);
                        dbg_print(0, MSG_ORIG(MSG_CFA_LLU), PREFIX,
                            EC_XWORD(oper1));

                        break;

                case 0x2f: /* GNU:DW_CFA_GNU_negative_offset_extended,reg,off */
                        oper1 = uleb_extract(&data[off], ndx);
                        soper = -uleb_extract(&data[off], ndx) *
                            state->ciedalign;
                        dbg_print(0, MSG_ORIG(MSG_CFA_CFAOFF), PREFIX,
                            REGNAME(oper1, rbuf1), EC_SXWORD(soper));
                        break;

                default:
                        /*
                         * Unrecognized OP code: DWARF data is variable length,
                         * so we don't know how many bytes to skip in order to
                         * advance to the next item. We cannot decode beyond
                         * this point, so dump the remainder in hex.
                         */
                        (*ndx)--;       /* Back up to unrecognized opcode */
```

```
                        dump_hex_bytes(data + off + *ndx, len - *ndx,
                            indent, 8, 1);
                        (*ndx) = len;
                        break;
                }
        }

#undef PREFIX
#undef REGNAME
#undef LOW_OP
}

void
dump_eh_frame(uchar_t *data, size_t datasize, uint64_t sh_addr,
    Half e_machine, uchar_t *e_ident, uint64_t gotaddr)
    Half e_machine, uchar_t *e_ident)
{
        Conv_dwarf_ehe_buf_t    dwarf_ehe_buf;
        dump_cfi_state_t        cfi_state;
        uint64_t        off, ndx;
        uint_t          cieid, cielength, cieversion, cieretaddr;
        int             ciePflag, cieZflag, cieLflag, cieLflag_present;
        uint_t          cieaugndx, length, id;
        char            *cieaugstr;

        cfi_state.e_machine = e_machine;
        cfi_state.e_ident = e_ident;
        cfi_state.sh_addr = sh_addr;
        cfi_state.do_swap = _elf_sys_encoding() != e_ident[EI_DATA];
        cfi_state.gotaddr = gotaddr;
#endif /* ! codereview */

        off = 0;
        while (off < datasize) {
                ndx = 0;

                /*
                 * Extract length in native format.  A zero length indicates
                 * that this CIE is a terminator and that processing for this
                 * unwind information should end. However, skip this entry and
                 * keep processing, just in case there is any other information
                 * remaining in this section.  Note, ld(1) will terminate the
                 * processing of the .eh_frame contents for this file after a
                 * zero length CIE, thus any information that does follow is
                 * ignored by ld(1), and is therefore questionable.
                 */
                length = (uint_t)dwarf_extract_uint(data + off, &ndx,
                    4, cfi_state.do_swap);
                if (length == 0) {
                        dbg_print(0, MSG_ORIG(MSG_UNW_ZEROTERM));
                        off += 4;
                        continue;
                }

                /*
                 * extract CIE id in native format
                 */
                id = (uint_t)dwarf_extract_uint(data + off, &ndx,
                    4, cfi_state.do_swap);

                /*
                 * A CIE record has an id of '0', otherwise this is a
                 * FDE entry and the 'id' is the CIE pointer.
                 */
                if (id == 0) {
                        uint64_t        persVal, ndx_save;
                        uint_t          axsize;

                        cielength = length;
                        cieid = id;
                        ciePflag = cfi_state.cieRflag = cieZflag = 0;
                        cieLflag = cieLflag_present = 0;

                        dbg_print(0, MSG_ORIG(MSG_UNW_CIE),
                            EC_XWORD(sh_addr + off));
                        dbg_print(0, MSG_ORIG(MSG_UNW_CIELNGTH),
                            cielength, cieid);

                        cieversion = data[off + ndx];
                        ndx += 1;
                        cieaugstr = (char *)(&data[off + ndx]);
                        ndx += strlen(cieaugstr) + 1;

                        dbg_print(0, MSG_ORIG(MSG_UNW_CIEVERS),
                            cieversion, cieaugstr);

                        cfi_state.ciecalign = uleb_extract(&data[off], &ndx);
                        cfi_state.ciedalign = sleb_extract(&data[off], &ndx);
                        cieretaddr = data[off + ndx];
                        ndx += 1;

                        dbg_print(0, MSG_ORIG(MSG_UNW_CIECALGN),
                            EC_XWORD(cfi_state.ciecalign),
                            EC_XWORD(cfi_state.ciedalign), cieretaddr);

                        if (cieaugstr[0])
                                dbg_print(0, MSG_ORIG(MSG_UNW_CIEAXVAL));

                        for (cieaugndx = 0; cieaugstr[cieaugndx]; cieaugndx++) {
                                switch (cieaugstr[cieaugndx]) {
                                case 'z':
                                        axsize = uleb_extract(&data[off], &ndx);
                                        dbg_print(0, MSG_ORIG(MSG_UNW_CIEAXSIZ),
                                            axsize);
                                        cieZflag = 1;
                                        /*
                                         * The auxiliary section can contain
                                         * unused padding bytes at the end, so
                                         * save the current index. Along with
                                         * axsize, we will use it to set ndx to
                                         * the proper continuation index after
                                         * the aux data has been processed.
                                         */
                                        ndx_save = ndx;
                                        break;
                                case 'P':
                                        ciePflag = data[off + ndx];
                                        ndx += 1;

                                        persVal = dwarf_ehe_extract(&data[off],
                                            &ndx, ciePflag, e_ident, B_FALSE,
                                            sh_addr, off + ndx, gotaddr);
                                            &ndx, ciePflag, e_ident,
                                            sh_addr, off + ndx);
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXPERS));
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXPERSENC),
                                            ciePflag, conv_dwarf_ehe(ciePflag,
                                            &dwarf_ehe_buf));
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXPERSRTN),
```

```
                                            EC_XWORD(persVal));
                                        break;
                                case 'R':
                                        cfi_state.cieRflag = data[off + ndx];
                                        ndx += 1;
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXCENC),
                                            cfi_state.cieRflag,
                                            conv_dwarf_ehe(cfi_state.cieRflag,
                                            &dwarf_ehe_buf));
                                        break;
                                case 'L':
                                        cieLflag_present = 1;
                                        cieLflag = data[off + ndx];
                                        ndx += 1;
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXLSDA),
                                            cieLflag, conv_dwarf_ehe(
                                            cieLflag, &dwarf_ehe_buf));
                                        break;
                                default:
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_CIEAXUNEC),
                                            cieaugstr[cieaugndx]);
                                        break;
                                }
                        }

                        /*
                         * If the z flag was present, reposition ndx using the
                         * length given. This will safely move us past any
                         * unaccessed padding bytes in the auxiliary section.
                         */
                        if (cieZflag)
                                ndx = ndx_save + axsize;

                        /*
                         * Any remaining data are Call Frame Instructions
                         */
                        if ((cielength + 4) > ndx)
                                dump_cfi(data, off, &ndx, cielength, &cfi_state,
                                    MSG_ORIG(MSG_UNW_CIECFI), 3);
                        off += cielength + 4;

                } else {
                        uint_t      fdelength = length;
                        int         fdecieptr = id;
                        uint64_t    fdeaddrrange;

                        dbg_print(0, MSG_ORIG(MSG_UNW_FDE),
                            EC_XWORD(sh_addr + off));
                        dbg_print(0, MSG_ORIG(MSG_UNW_FDELNGTH),
                            fdelength, fdecieptr);

                        cfi_state.fdeinitloc = dwarf_ehe_extract(&data[off],
                            &ndx, cfi_state.cieRflag, e_ident, B_FALSE,
                            sh_addr, off + ndx, gotaddr);
                            &ndx, cfi_state.cieRflag, e_ident,
                            sh_addr, off + ndx);
                        fdeaddrrange = dwarf_ehe_extract(&data[off], &ndx,
                            (cfi_state.cieRflag & ~DW_EH_PE_pcrel),
                            e_ident, B_FALSE, sh_addr, off + ndx, gotaddr);
                            e_ident, sh_addr, off + ndx);

                        dbg_print(0, MSG_ORIG(MSG_UNW_FDEINITLOC),
                            EC_XWORD(cfi_state.fdeinitloc),
                                EC_XWORD(fdeaddrrange),
                                EC_XWORD(cfi_state.fdeinitloc + fdeaddrrange - 1));

                        if (cieaugstr[0])
                                dbg_print(0, MSG_ORIG(MSG_UNW_FDEAXVAL));
                        if (cieZflag) {
                                uint64_t    val;
                                uint64_t    lndx;

                                val = uleb_extract(&data[off], &ndx);
                                lndx = ndx;
                                ndx += val;
                                dbg_print(0, MSG_ORIG(MSG_UNW_FDEAXSIZE),
                                    EC_XWORD(val));
                                if (val && cieLflag_present) {
                                        uint64_t    lsda;

                                        lsda = dwarf_ehe_extract(&data[off],
                                            &lndx, cieLflag, e_ident,
                                            B_FALSE, sh_addr, off + lndx,
                                            gotaddr);
                                            sh_addr, off + lndx);
                                        dbg_print(0,
                                            MSG_ORIG(MSG_UNW_FDEAXLSDA),
                                            EC_XWORD(lsda));
                                }
                        }
                        if ((fdelength + 4) > ndx)
                                dump_cfi(data, off, &ndx, fdelength, &cfi_state,
                                    MSG_ORIG(MSG_UNW_FDECFI), 6);
                        off += fdelength + 4;
                }
        }
}
```
______***unchanged_portion_omitted_***

```
**********************************************************
  144983 Thu Oct 11 22:38:18 2012
new/usr/src/cmd/sgs/elfdump/common/elfdump.c
3265 link-editor builds bogus .eh_frame_hdr on ia32
**********************************************************
______unchanged_portion_omitted_

/*
 * Display the contents of GNU/amd64 .eh_frame and .eh_frame_hdr
 * sections.
 *
 * entry:
 *      cache - Cache of all section headers
 *      shndx - Index of .eh_frame or .eh_frame_hdr section to be displayed
 *      uphdr - NULL, or unwind program header associated with
 *              the .eh_frame_hdr section.
 *      ehdr - ELF header for file
 *      eh_state - Data used across calls to this routine. The
 *              caller should zero it before the first call, and
 *              pass it on every call.
 *      osabi - OSABI to use in displaying information
 *      file - Name of file
 *      flags - Command line option flags
 */
static void
unwind_eh_frame(Cache *cache, Word shndx, Word shnum, Phdr *uphdr, Ehdr *ehdr,
unwind_eh_frame(Cache *cache, Word shndx, Phdr *uphdr, Ehdr *ehdr,
    gnu_eh_state_t *eh_state, uchar_t osabi, const char *file, uint_t flags)
{
#if     defined(_ELF64)
#define MSG_UNW_BINSRTAB2       MSG_UNW_BINSRTAB2_64
#define MSG_UNW_BINSRTABENT     MSG_UNW_BINSRTABENT_64
#else
#define MSG_UNW_BINSRTAB2       MSG_UNW_BINSRTAB2_32
#define MSG_UNW_BINSRTABENT     MSG_UNW_BINSRTABENT_32
#endif

        Cache                   *_cache = &cache[shndx];
        Shdr                    *shdr = _cache->c_shdr;
        uchar_t                 *data = (uchar_t *)(_cache->c_data->d_buf);
        size_t                  datasize = _cache->c_data->d_size;
        Conv_dwarf_ehe_buf_t    dwarf_ehe_buf;
        uint64_t                ndx, frame_ptr, fde_cnt, tabndx;
        uint_t                  vers, frame_ptr_enc, fde_cnt_enc, table_enc;
        uint64_t                initloc, initloc0;
        uint64_t                gotaddr = 0;
        int                     cnt;
#endif /* ! codereview */

        for (cnt = 1; cnt < shnum; cnt++) {
                if (strncmp(cache[cnt].c_name, MSG_ORIG(MSG_ELF_GOT),
                    MSG_ELF_GOT_SIZE) == 0) {
                        gotaddr = cache[cnt].c_shdr->sh_addr;
                        break;
                }
        }
#endif /* ! codereview */

        /*
         * Is this a .eh_frame_hdr?
         */
        if ((uphdr && (shdr->sh_addr == uphdr->p_vaddr)) ||
            (strncmp(_cache->c_name, MSG_ORIG(MSG_SCN_FRMHDR),
            MSG_SCN_FRMHDR_SIZE) == 0)) {
                /*
                 * There can only be a single .eh_frame_hdr.
```

```
                 * Flag duplicates.
                 */
                if (++eh_state->hdr_cnt > 1)
                        (void) fprintf(stderr, MSG_INTL(MSG_ERR_MULTEHFRMHDR),
                            file, EC_WORD(shndx), _cache->c_name);

                dbg_print(0, MSG_ORIG(MSG_UNW_FRMHDR));
                ndx = 0;

                vers = data[ndx++];
                frame_ptr_enc = data[ndx++];
                fde_cnt_enc = data[ndx++];
                table_enc = data[ndx++];

                dbg_print(0, MSG_ORIG(MSG_UNW_FRMVERS), vers);

                frame_ptr = dwarf_ehe_extract(data, &ndx, frame_ptr_enc,
                    ehdr->e_ident, B_TRUE, shdr->sh_addr, ndx, gotaddr);
                    ehdr->e_ident, shdr->sh_addr, ndx);
                if (eh_state->hdr_cnt == 1) {
                        eh_state->hdr_ndx = shndx;
                        eh_state->frame_ptr = frame_ptr;
                }

                dbg_print(0, MSG_ORIG(MSG_UNW_FRPTRENC),
                    conv_dwarf_ehe(frame_ptr_enc, &dwarf_ehe_buf),
                    EC_XWORD(frame_ptr));

                fde_cnt = dwarf_ehe_extract(data, &ndx, fde_cnt_enc,
                    ehdr->e_ident, B_TRUE, shdr->sh_addr, ndx, gotaddr);
                    ehdr->e_ident, shdr->sh_addr, ndx);

                dbg_print(0, MSG_ORIG(MSG_UNW_FDCNENC),
                    conv_dwarf_ehe(fde_cnt_enc, &dwarf_ehe_buf),
                    EC_XWORD(fde_cnt));
                dbg_print(0, MSG_ORIG(MSG_UNW_TABENC),
                    conv_dwarf_ehe(table_enc, &dwarf_ehe_buf));
                dbg_print(0, MSG_ORIG(MSG_UNW_BINSRTAB1));
                dbg_print(0, MSG_ORIG(MSG_UNW_BINSRTAB2));

                for (tabndx = 0; tabndx < fde_cnt; tabndx++) {
                        initloc = dwarf_ehe_extract(data, &ndx, table_enc,
                            ehdr->e_ident, B_TRUE, shdr->sh_addr, ndx, gotaddr);
                            ehdr->e_ident, shdr->sh_addr, ndx);
                        /*LINTED:E_VAR_USED_BEFORE_SET*/
                        if ((tabndx != 0) && (initloc0 > initloc))
                                (void) fprintf(stderr,
                                    MSG_INTL(MSG_ERR_BADSORT), file,
                                    _cache->c_name, EC_WORD(tabndx));
                        dbg_print(0, MSG_ORIG(MSG_UNW_BINSRTABENT),
                            EC_XWORD(initloc),
                            EC_XWORD(dwarf_ehe_extract(data, &ndx,
                            table_enc, ehdr->e_ident, B_TRUE, shdr->sh_addr,
                            ndx, gotaddr)));
                            table_enc, ehdr->e_ident, shdr->sh_addr,
                            ndx)));
                        initloc0 = initloc;
                }
        } else {                /* Display the .eh_frame section */
                eh_state->frame_cnt++;
                if (eh_state->frame_cnt == 1) {
                        eh_state->frame_ndx = shndx;
                        eh_state->frame_base = shdr->sh_addr;
                } else if ((eh_state->frame_cnt >  1) &&
                    (ehdr->e_type != ET_REL)) {
                        Conv_inv_buf_t  inv_buf;
                        (void) fprintf(stderr, MSG_INTL(MSG_WARN_MULTEHFRM),
                            file, EC_WORD(shndx), _cache->c_name,
                            conv_ehdr_type(osabi, ehdr->e_type, 0, &inv_buf));
                }
                dump_eh_frame(data, datasize, shdr->sh_addr,
                    ehdr->e_machine, ehdr->e_ident, gotaddr);
                    ehdr->e_machine, ehdr->e_ident);
        }

        /*
         * If we've seen the .eh_frame_hdr and the first .eh_frame section,
         * compare the header frame_ptr to the address of the actual frame
         * section to ensure the link-editor got this right.  Note, this
         * diagnostic is only produced when unwind information is explicitly
         * asked for, as shared objects built with an older ld(1) may reveal
         * this inconsistency.  Although an inconsistency, it doesn't seem to
         * have any adverse effect on existing tools.
         */
        if (((flags & FLG_MASK_SHOW) != FLG_MASK_SHOW) &&
            (eh_state->hdr_cnt > 0) && (eh_state->frame_cnt > 0) &&
            (eh_state->frame_ptr != eh_state->frame_base))
                (void) fprintf(stderr, MSG_INTL(MSG_ERR_BADEHFRMPTR),
                    file, EC_WORD(eh_state->hdr_ndx),
                    cache[eh_state->hdr_ndx].c_name,
                    EC_XWORD(eh_state->frame_ptr),
                    EC_WORD(eh_state->frame_ndx),
                    cache[eh_state->frame_ndx].c_name,
                    EC_XWORD(eh_state->frame_base));
#undef MSG_UNW_BINSRTAB2
#undef MSG_UNW_BINSRTABENT
}
```

______***unchanged_portion_omitted_***

```
/*
 * Display information from unwind/exception sections:
 *
 * -    GNU/amd64 .eh_frame and .eh_frame_hdr
 * -    Sun C++ .exception_ranges
 *
 */
static void
unwind(Cache *cache, Word shnum, Word phnum, Ehdr *ehdr, uchar_t osabi,
    const char *file, Elf *elf, uint_t flags)
{
        static Word phdr_types[] = { PT_SUNW_UNWIND, PT_SUNW_EH_FRAME };

        Word                    cnt;
        Phdr                    *uphdr = NULL;
        gnu_eh_state_t          eh_state;

        /*
         * Historical background: .eh_frame and .eh_frame_hdr sections
         * come from the GNU compilers (particularly C++), and are used
         * under all architectures. Their format is based on DWARF. When
         * the amd64 ABI was defined, these sections were adopted wholesale
         * from the existing practice.
         *
         * When amd64 support was added to Solaris, support for these
         * sections was added, using the SHT_AMD64_UNWIND section type
         * to identify them. At first, we ignored them in objects for
         * non-amd64 targets, but later broadened our support to include
         * other architectures in order to better support gcc-generated
         * objects.
         *
         * .exception_ranges implement the same basic concepts, but
```

```
         * were invented at Sun for the Sun C++ compiler.
         *
         * We match these sections by name, rather than section type,
         * because they can come in as either SHT_AMD64_UNWIND, or as
         * SHT_PROGBITS, and because the type isn't enough to determine
         * how they should be interpreted.
         */
        /* Find the program header for .eh_frame_hdr if present */
        if (phnum)
                uphdr = getphdr(phnum, phdr_types,
                    sizeof (phdr_types) / sizeof (*phdr_types), file, elf);

        /*
         * eh_state is used to retain data used by unwind_eh_frame()
         * across calls.
         */
        bzero(&eh_state, sizeof (eh_state));

        for (cnt = 1; cnt < shnum; cnt++) {
                Cache           *_cache = &cache[cnt];
                Shdr            *shdr = _cache->c_shdr;
                int             is_exrange;

                /*
                 * Skip sections of the wrong type. On amd64, they
                 * can be SHT_AMD64_UNWIND. On all platforms, they
                 * can be SHT_PROGBITS (including amd64, if using
                 * the GNU compilers).
                 *
                 * Skip anything other than these two types. The name
                 * test below will thin out the SHT_PROGBITS that don't apply.
                 */
                if ((shdr->sh_type != SHT_PROGBITS) &&
                    (shdr->sh_type != SHT_AMD64_UNWIND))
                        continue;

                /*
                 * Only sections with certain well known names are of interest.
                 * These are:
                 *
                 *      .eh_frame - amd64/GNU-compiler unwind sections
                 *      .eh_frame_hdr - Sorted table referencing .eh_frame
                 *      .exception_ranges - Sun C++ unwind sections
                 *
                 * We do a prefix comparison, allowing for naming conventions
                 * like .eh_frame.foo, hence the use of strncmp() rather than
                 * strcmp(). This means that we only really need to test for
                 * .eh_frame, as it's a prefix of .eh_frame_hdr.
                 */
                is_exrange =  strncmp(_cache->c_name,
                    MSG_ORIG(MSG_SCN_EXRANGE), MSG_SCN_EXRANGE_SIZE) == 0;
                if ((strncmp(_cache->c_name, MSG_ORIG(MSG_SCN_FRM),
                    MSG_SCN_FRM_SIZE) != 0) && !is_exrange)
                        continue;

                if (!match(MATCH_F_ALL, _cache->c_name, cnt, shdr->sh_type))
                        continue;

                if (_cache->c_data == NULL)
                        continue;

                dbg_print(0, MSG_ORIG(MSG_STR_EMPTY));
                dbg_print(0, MSG_INTL(MSG_ELF_SCN_UNWIND), _cache->c_name);

                if (is_exrange)
                        unwind_exception_ranges(_cache, file,
				    _elf_sys_encoding() != ehdr->e_ident[EI_DATA]);
			else
				unwind_eh_frame(cache, cnt, shnum, uphdr, ehdr,
				    &eh_state, osabi, file, flags);
				unwind_eh_frame(cache, cnt, uphdr, ehdr, &eh_state,
				    osabi, file, flags);
		}
}
```
______***unchanged_portion_omitted_***

```
**********************************************************
    9285 Thu Oct 11 22:38:19 2012
new/usr/src/cmd/sgs/include/dwarf.h
3265 link-editor builds bogus .eh_frame_hdr on ia32
**********************************************************
/*
 * 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) 1989 AT&T */
/*        All Rights Reserved   */


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

#ifndef _DWARF_H
#define _DWARF_H

#include <sys/types.h>


#ifdef  __cplusplus
extern "C" {
#endif



/* dwarf.h - manifest constants used in the .debug section of ELF files */


/* the "tag" - the first short of any legal record */

#define TAG_padding                     0x0000
#define TAG_array_type                  0x0001
#define TAG_class_type                  0x0002
#define TAG_entry_point                 0x0003
#define TAG_enumeration_type            0x0004
#define TAG_formal_parameter            0x0005
#define TAG_global_subroutine           0x0006
#define TAG_global_variable             0x0007
#define TAG_imported_declaration        0x0008
#define TAG_inline_subroutine           0x0009
#define TAG_label                       0x000a
#define TAG_lexical_block               0x000b
#define TAG_local_variable              0x000c
#define TAG_member                      0x000d
#define TAG_member_function             0x000e
```

```
#define TAG_pointer_type                0x000f
#define TAG_reference_type              0x0010
#define TAG_source_file                 0x0011
#define TAG_string_type                 0x0012
#define TAG_structure_type              0x0013
#define TAG_subroutine                  0x0014
#define TAG_subroutine_type             0x0015
#define TAG_typedef                     0x0016
#define TAG_union_type                  0x0017
#define TAG_unspecified_parameters      0x0018
#define TAG_variant                     0x0019


/* attribute forms are encoded as part */
/* of the attribute name and must fit */
/* into 4 bits */

#define FORM_MASK       0xf

#define FORM_NONE       0x0     /* error */
#define FORM_ADDR       0x1     /* relocated address */
#define FORM_REF        0x2     /* reference to another .debug entry */
#define FORM_BLOCK2     0x3     /* block with 2-byte length */
#define FORM_BLOCK4     0x4     /* block with 4-byte length (unused) */
#define FORM_DATA2      0x5     /* 2 bytes */
#define FORM_DATA4      0x6     /* 4 bytes */
#define FORM_DATA8      0x7     /* 8 bytes (two 4-byte values) */
#define FORM_STRING     0x8     /* NUL-terminated string */


/* attribute names, halfwords with low 4 bits indicating the form */

#define AT_padding       (0x0000|FORM_NONE)     /* just padding */
#define AT_sibling       (0x0010|FORM_REF)      /* next owned declaration */
#define AT_location      (0x0020|FORM_BLOCK2)   /* location description */
#define AT_name          (0x0030|FORM_STRING)   /* symbol name */
#define AT_dimensions    (0x0040|FORM_DATA2)    /* array dimensions */
#define AT_fund_type     (0x0050|FORM_DATA2)    /* fund type enum */
#define AT_mod_fund_type (0x0060|FORM_BLOCK2)   /* modifiers & fund type enum */
#define AT_user_def_type (0x0070|FORM_REF)      /* type entry */
#define AT_mod_u_d_type  (0x0080|FORM_BLOCK2)   /* modifiers & type entry ref */
#define AT_ordering      (0x0090|FORM_DATA2)    /* array row/column major */
#define AT_subscr_data   (0x00a0|FORM_BLOCK2)   /* list of array dim info */
#define AT_byte_size     (0x00b0|FORM_DATA4)    /* number bytes per instance */
#define AT_bit_offset    (0x00c0|FORM_DATA2)    /* number bits padding */
#define AT_bit_size      (0x00d0|FORM_DATA4)    /* number bits per instance */
#define AT_deriv_list    (0x00e0|FORM_BLOCK2)   /* list of base class refs */
#define AT_element_list  (0x00f0|FORM_BLOCK4)   /* list of enum data elements */
#define AT_stmt_list     (0x0100|FORM_DATA4)    /* offset in .line sect */
#define AT_low_pc        (0x0110|FORM_ADDR)     /* first machine instr */
#define AT_high_pc       (0x0120|FORM_ADDR)     /* beyond last machine instr */
#define AT_language      (0x0130|FORM_DATA4)    /* compiler enumeration */
#define AT_member        (0x0140|FORM_REF)      /* class description */
#define AT_discr         (0x0150|FORM_REF)      /* discriminant entry */
#define AT_discr_value   (0x0160|FORM_BLOCK2)   /* value of discr */
#define AT_visibility    (0x0170|FORM_DATA2)    /* visibility enumeration */
#define AT_import        (0x0180|FORM_REF)      /* imported declaration */
#define AT_string_length (0x0190|FORM_BLOCK2)   /* runtime string size */


/* atoms which compose a location description; must fit in a byte */

#define OP_UNK          0x00    /* error */
#define OP_REG          0x01    /* push register (number) */
#define OP_BASEREG      0x02    /* push value of register (number) */
#define OP_ADDR         0x03    /* push address (relocated address) */
```

```
#define OP_CONST        0x04    /* push constant (number) */
#define OP_DEREF2       0x05    /* pop, deref and push 2 bytes (as a long) */
#define OP_DEREF4       0x06    /* pop, deref and push 4 bytes (as a long) */
#define OP_ADD          0x07    /* pop top 2 items, add, push result */

/* fundamental types; must fit in two bytes */

#define FT_none                 0x0000  /* error */
#define FT_char                 0x0001  /* "plain" char */
#define FT_signed_char          0x0002
#define FT_unsigned_char        0x0003
#define FT_short                0x0004  /* "plain" short */
#define FT_signed_short         0x0005
#define FT_unsigned_short       0x0006
#define FT_integer              0x0007  /* "plain" integer */
#define FT_signed_integer       0x0008
#define FT_unsigned_integer     0x0009
#define FT_long                 0x000a  /* "plain" long */
#define FT_signed_long          0x000b
#define FT_unsigned_long        0x000c
#define FT_pointer              0x000d  /* (void *) */
#define FT_float                0x000e
#define FT_dbl_prec_float       0x000f
#define FT_ext_prec_float       0x0010
#define FT_complex              0x0011
#define FT_dbl_prec_complex     0x0012
#define FT_set                  0x0013
#define FT_void                 0x0014


/* type modifiers; must fit in a byte */

#define MOD_none                0x00    /* error */
#define MOD_pointer_to          0x01
#define MOD_reference_to        0x02


/* the "format" byte for array descriptions; formed from three */
/* one-bit fields */

#define FMT_FT  0               /* fundamental type */
#define FMT_UDT 1               /* user-defined type */

#define FMT_CONST       0       /* 4-byte constant */
#define FMT_EXPR        1       /* block with 2-byte length (loc descr) */

#define FMT_FT_C_C      ((FMT_FT <<2) | (FMT_CONST<<1) | (FMT_CONST))
#define FMT_FT_C_X      ((FMT_FT <<2) | (FMT_CONST<<1) | (FMT_EXPR))
#define FMT_FT_X_C      ((FMT_FT <<2) | (FMT_EXPR <<1) | (FMT_CONST))
#define FMT_FT_X_X      ((FMT_FT <<2) | (FMT_EXPR <<1) | (FMT_EXPR))
#define FMT_UT_C_C      ((FMT_UDT<<2) | (FMT_CONST<<1) | (FMT_CONST))
#define FMT_UT_C_X      ((FMT_UDT<<2) | (FMT_CONST<<1) | (FMT_EXPR))
#define FMT_UT_X_C      ((FMT_UDT<<2) | (FMT_EXPR <<1) | (FMT_CONST))
#define FMT_UT_X_X      ((FMT_UDT<<2) | (FMT_EXPR <<1) | (FMT_EXPR))

#define FMT_ET          8       /* element type */


/* ordering of arrays */

#define ORD_row_major   0
#define ORD_col_major   1


/* visibility values */
```

```
#define VIS_local       0       /* for static functions in C */
#define VIS_exported    1       /* for Modula */

/*
 * DWARF Exception Header Encoding
 *
 * The DWARF Exception Header Encoding is used to describe the type of data
 * used in the .eh_frame_hdr section. The upper 4 bits indicate how the value
 * is to be applied. The lower 4 bits indicate the format of the data.
 */

/*
 * Dwarf Exception Header Value format
 */
#define DW_EH_PE_omit           0xff    /* No value is present. */
#define DW_EH_PE_absptr         0x00    /* Value is a void* */
#define DW_EH_PE_uleb128        0x01    /* Unsigned value is encoded using */
                                        /*   the Little Endian */
                                        /*   Base 128 (LEB128) */
#define DW_EH_PE_udata2         0x02    /* A 2 bytes unsigned value. */
#define DW_EH_PE_udata4         0x03    /* A 4 bytes unsigned value. */
#define DW_EH_PE_udata8         0x04    /* An 8 bytes unsigned value. */
#define DW_EH_PE_signed         0x08    /* bit on for all signed encodings */
#define DW_EH_PE_sleb128        0x09    /* Signed value is encoded using */
                                        /*   the Little Endian */
                                        /*   Base 128 (LEB128) */
#define DW_EH_PE_sdata2         0x0a    /* A 2 bytes signed value. */
#define DW_EH_PE_sdata4         0x0b    /* A 4 bytes signed value. */
#define DW_EH_PE_sdata8         0x0c    /* An 8 bytes signed value. */

/*
 * Dwarf Exception Header application
 */
#define DW_EH_PE_absptr         0x00    /* Value is used with no */
                                        /*  modification. */
#define DW_EH_PE_pcrel          0x10    /* Value is reletive to the location */
                                        /*  of itself */
#define DW_EH_PE_textrel        0x20
#define DW_EH_PE_datarel        0x30    /* Value is reletive to the beginning */
                                        /*  of the eh_frame_hdr segment */
                                        /*  ( segment type PT_AMD64_UNWIND ) */
                                        /*  when within that segment, or to */
                                        /*  the GOT when without. */
#endif /* ! codereview */
#define DW_EH_PE_funcrel        0x40
#define DW_EH_PE_aligned        0x50    /* value is an aligned void* */
#define DW_EH_PE_indirect       0x80    /* bit to signal indirection after */
                                        /*  relocation */


/* language/compiler enumeration */

typedef enum _LANG {
        LANG_UNK = 0,
        LANG_ANSI_C_V1 = 1
} LANG;

/*
 * Little Endian Base 128 (leb128) encoding/decoding routines
 */
extern  uint64_t        uleb_extract(unsigned char *, uint64_t *);
extern  int64_t         sleb_extract(unsigned char *, uint64_t *);
extern  uint64_t        dwarf_ehe_extract(unsigned char *, uint64_t *,
                            uint_t, unsigned char *, boolean_t, uint64_t,
                            uint64_t, uint64_t);
                            uint_t, unsigned char *, uint64_t, uint64_t);
```

```
#ifdef  __cplusplus
}
```
______***unchanged_portion_omitted***_

****
   22262 Thu Oct 11 22:38:19 2012
new/usr/src/cmd/sgs/libld/common/unwind.c
3265 link-editor builds bogus .eh_frame_hdr on ia32
****
______unchanged_portion_omitted_

```
uintptr_t
ld_unwind_populate_hdr(Ofl_desc *ofl)
{
        uchar_t         *hdrdata;
        uint_t          *binarytable;
        uint_t          hdroff;
        Aliste          idx;
        Addr            hdraddr;
        Os_desc         *hdrosp;
        Os_desc         *osp;
        Os_desc         *first_unwind;
        uint_t          fde_count;
        uint_t          *uint_ptr;
        int             bswap = (ofl->ofl_flags1 & FLG_OF1_ENCDIFF) != 0;

        /*
         * Are we building the unwind hdr?
         */
        if ((hdrosp = ofl->ofl_unwindhdr) == 0)
                return (1);

        hdrdata = hdrosp->os_outdata->d_buf;
        hdraddr = hdrosp->os_shdr->sh_addr;
        hdroff = 0;

        /*
         * version == 1
         */
        hdrdata[hdroff++] = 1;
        /*
         * The encodings are:
         *
         *  eh_frameptr_enc     sdata4 | pcrel
         *  fde_count_enc       udata4
         *  table_enc           sdata4 | datarel
         */
        hdrdata[hdroff++] = DW_EH_PE_sdata4 | DW_EH_PE_pcrel;
        hdrdata[hdroff++] = DW_EH_PE_udata4;
        hdrdata[hdroff++] = DW_EH_PE_sdata4 | DW_EH_PE_datarel;

        /*
         *      Header Offsets
         *      -----------------------------------
         *      byte        version             +1
         *      byte        eh_frame_ptr_enc    +1
         *      byte        fde_count_enc       +1
         *      byte        table_enc           +1
         *      4 bytes     eh_frame_ptr        +4
         *      4 bytes     fde_count           +4
         */
        /* LINTED */
        binarytable =  (uint_t *)(hdrdata + 12);
        first_unwind = 0;
        fde_count = 0;

        for (APLIST_TRAVERSE(ofl->ofl_unwind, idx, osp)) {
                uchar_t         *data;
                size_t          size;
                uint64_t        off = 0;
```

```
                uint_t          cieRflag = 0, ciePflag = 0;
                Shdr            *shdr;

                /*
                 * remember first UNWIND section to
                 * point to in the frame_ptr entry.
                 */
                if (first_unwind == 0)
                        first_unwind = osp;

                data = osp->os_outdata->d_buf;
                shdr = osp->os_shdr;
                size = shdr->sh_size;

                while (off < size) {
                        uint_t      length, id;
                        uint64_t    ndx = 0;

                        /*
                         * Extract length in lsb format.  A zero length
                         * indicates that this CIE is a terminator and that
                         * processing of unwind information is complete.
                         */
                        length = extract_uint(data + off, &ndx, bswap);
                        if (length == 0)
                                goto done;

                        /*
                         * Extract CIE id in lsb format.
                         */
                        id = extract_uint(data + off, &ndx, bswap);

                        /*
                         * A CIE record has a id of '0'; otherwise
                         * this is a FDE entry and the 'id' is the
                         * CIE pointer.
                         */
                        if (id == 0) {
                                char    *cieaugstr;
                                uint_t  cieaugndx;

                                ciePflag = 0;
                                cieRflag = 0;
                                /*
                                 * We need to drill through the CIE
                                 * to find the Rflag.  It's the Rflag
                                 * which describes how the FDE code-pointers
                                 * are encoded.
                                 */

                                /*
                                 * burn through version
                                 */
                                ndx++;

                                /*
                                 * augstr
                                 */
                                cieaugstr = (char *)(&data[off + ndx]);
                                ndx += strlen(cieaugstr) + 1;

                                /*
                                 * calign & dalign
                                 */
                                (void) uleb_extract(&data[off], &ndx);
                                (void) sleb_extract(&data[off], &ndx);
                                /*
                                 * retreg
                                 */
                                ndx++;

                                /*
                                 * we walk through the augmentation
                                 * section now looking for the Rflag
                                 */
                                for (cieaugndx = 0; cieaugstr[cieaugndx];
                                    cieaugndx++) {
                                        /* BEGIN CSTYLED */
                                        switch (cieaugstr[cieaugndx]) {
                                        case 'z':
                                            /* size */
                                            (void) uleb_extract(&data[off],
                                                &ndx);
                                            break;
                                        case 'P':
                                            /* personality */
                                            ciePflag = data[off + ndx];
                                            ndx++;
                                                /*
                                                 * Just need to extract the
                                                 * value to move on to the next
                                                 * field.
                                                 */
                                            (void) dwarf_ehe_extract(
                                                &data[off + ndx],
                                                &ndx, ciePflag,
                                                ofl->ofl_dehdr->e_ident, B_FALSE
                                                shdr->sh_addr, off + ndx, 0);
                                                ofl->ofl_dehdr->e_ident,
                                                shdr->sh_addr, off + ndx);
                                            break;
                                        case 'R':
                                            /* code encoding */
                                            cieRflag = data[off + ndx];
                                            ndx++;
                                            break;
                                        case 'L':
                                            /* lsda encoding */
                                            ndx++;
                                            break;
                                        }
                                        /* END CSTYLED */
                                }
                        } else {
                                uint_t      bintabndx;
                                uint64_t    initloc;
                                uint64_t    fdeaddr;
                                uint64_t    gotaddr = 0;

                                if (ofl->ofl_osgot != NULL)
                                        gotaddr =
                                            ofl->ofl_osgot->os_shdr->sh_addr;
#endif /* ! codereview */

                                initloc = dwarf_ehe_extract(&data[off],
                                    &ndx, cieRflag, ofl->ofl_dehdr->e_ident,
                                    B_FALSE,
                                    shdr->sh_addr, off + ndx,
                                    gotaddr);
                                    shdr->sh_addr, off + ndx);
```

```
                                /*
                                 * Ignore FDEs with initloc set to 0.
                                 * initloc will not be 0 unless this FDE was
                                 * abandoned due to GNU linkonce processing.
                                 * The 0 value occurs because we don't resolve
                                 * sloppy relocations for unwind header target
                                 * sections.
                                 */
                                if (initloc != 0) {
                                        bintabndx = fde_count * 2;
                                        fde_count++;

                                        /*
                                         * FDEaddr is adjusted
                                         * to account for the length & id which
                                         * have already been consumed.
                                         */
                                        fdeaddr = shdr->sh_addr + off;

                                        binarytable[bintabndx] =
                                            (uint_t)(initloc - hdraddr);
                                        binarytable[bintabndx + 1] =
                                            (uint_t)(fdeaddr - hdraddr);
                                }
                        }

                        /*
                         * the length does not include the length
                         * itself - so account for that too.
                         */
                        off += length + 4;
                }
        }

done:
        /*
         * Do a quicksort on the binary table. If this is a cross
         * link from a system with the opposite byte order, xlate
         * the resulting values into LSB order.
         */
        framehdr_addr = hdraddr;
        qsort((void *)binarytable, (size_t)fde_count,
            (size_t)(sizeof (uint_t) * 2), bintabcompare);
        if (bswap) {
                uint_t  *btable = binarytable;
                uint_t  cnt;

                for (cnt = fde_count * 2; cnt-- > 0; btable++)
                        *btable = ld_bswap_Word(*btable);
        }

        /*
         * Fill in:
         *      first_frame_ptr
         *      fde_count
         */
        hdroff = 4;
        /* LINTED */
        uint_ptr = (uint_t *)(&hdrdata[hdroff]);
        *uint_ptr = first_unwind->os_shdr->sh_addr -
            (hdrosp->os_shdr->sh_addr + hdroff);
        if (bswap)
                *uint_ptr = ld_bswap_Word(*uint_ptr);

        hdroff += 4;
        /* LINTED */
```

```
	uint_ptr = (uint_t *)&hdrdata[hdroff];
	*uint_ptr = fde_count;
	if (bswap)
		*uint_ptr = ld_bswap_Word(*uint_ptr);

	/*
	 * If relaxed relocations are active, then there is a chance
	 * that we didn't use all the space reserved for this section.
	 * For details, see the note at head of ld_unwind_make_hdr() above.
	 *
	 * Find the PT_SUNW_UNWIND program header, and change the size values
	 * to the size of the subset of the section that was actually used.
	 */
	if (ofl->ofl_flags1 & FLG_OF1_RLXREL) {
		Word	phnum = ofl->ofl_nehdr->e_phnum;
		Phdr	*phdr = ofl->ofl_phdr;

		for (; phnum-- > 0; phdr++) {
			if (phdr->p_type == PT_SUNW_UNWIND) {
				phdr->p_memsz = 12 + (8 * fde_count);
				phdr->p_filesz = phdr->p_memsz;
				break;
			}
		}
	}

	return (1);
}
```
______***unchanged_portion_omitted_***

```
**********************************************************
   86912 Thu Oct 11 22:38:19 2012
new/usr/src/cmd/sgs/packages/common/SUNWonld-README
3265 link-editor builds bogus .eh_frame_hdr on ia32
**********************************************************
#
# Copyright (c) 1996, 2010, Oracle and/or its affiliates. All rights reserved.
#
# 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
#
# Note: The contents of this file are used to determine the versioning
#       information for the SGS toolset. The number of CRs listed in
#       this file must grow monotonically, or the SGS version will
#       move backwards, causing a great deal of confusion. As such,
#       CRs must never be removed from this file. See
#       libconv/common/bld_vernote.ksh, and bug#4519569 for more
#       details on SGS versioning.
#
--------------------------------------------------------------------------------
SUNWonld - link-editors development package.
--------------------------------------------------------------------------------

  The SUNWonld package is an internal development package containing the
  link-editors and some related tools.  All components live in the OSNET
  source base, but not all components are delivered as part of the normal
  OSNET consolidation.  The intent of this package is to provide access
  to new features/bugfixes before they become generally available.

  General link-editor information can be found:

    http://linkers.central/
    http://linkers.sfbay/       (also known as linkers.eng)

  Comments and Questions:

    Contact Rod Evans, Ali Bahrami, and/or Seizo Sakurai.

  Warnings:

    The postremove script for this package employs /usr/sbin/static/mv,
    and thus, besides the common core dependencies, this package also
    has a dependency on the SUNWsutl package.

  Patches:

    If the patch has been made official, you'll find it in:

        http://sunsolve.east/cgi/show.pl?target=patches/os-patches
```

```
    If it hasn't been released, the patch will be in:

        /net/sunsoftpatch/patches/temporary

    Note, any patches logged here refer to the temporary ("T") name, as we
    never know when they're made official, and although we try to keep all
    patch information up-to-date the real status of any patch can be
    determined from:

        http://sunsoftpatch.eng

    If it has been obsoleted, the patch will be in:

        /net/on${RELEASE}-patch/on${RELEASE}/patches/${MACH}/obsolete


  History:

    Note, starting after Solaris 10, letter codes in parenthesis may
    be found following the bug synopsis. Their meanings are as follows:

        (D) A documentation change accompanies the implementation change.
        (P) A packaging change accompanies the implementation change.

    In all cases, see the implementation bug report for details.

    The following bug fixes exist in the OSNET consolidation workspace
    from which this package is created:

---------
Solaris 8
---------
Bugid   Risk Synopsis
================================================================================
4225937 i386 linker emits sparc specific warning messages
4215164 shf_order flag handling broken by fix for 4194028.
4215587 using ld and the -r option on solaris 7 with compiler option -xarch=v9
        causes link errors.
4234657 103627-08 breaks purify 4.2 (plt padding should not be enabled for
        32-bit)
4235241 dbx no longer gets dlclose notification.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-05  (never released)
        Solaris/SunOS 5.7_x86           patch 106951-05  (never released)
        Solaris/SunOS 5.6_sparc         patch 107733-02  (never released)
        Solaris/SunOS 5.6_x86           patch 107734-02
--------------------------------------------------------------------------------
4248290 inetd dumps core upon bootup - failure in dlclose() logic.
4238071 dlopen() leaks while descriptors under low memory conditions
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-06
        Solaris/SunOS 5.7_x86           patch 106951-06
        Solaris/SunOS 5.6_sparc         patch 107733-03  (never released)
        Solaris/SunOS 5.6_x86           patch 107734-03
--------------------------------------------------------------------------------
4267980 INITFIRST flag of the shard object could be ignored.
--------------------------------------------------------------------------------
All the above changes plus:
    4238973 fix for 4121152 affects linking of Ada objects
    4158744 patch 103627-02 causes core when RPATH has blank entry and
            dlopen/dlclose is used
are incorporated in the following patches:
        Solaris/SunOS 5.5.1_sparc       patch 103627-12  (never released)
        Solaris/SunOS 5.5.1_x86         patch 103628-11
```

```
--------------------------------------------------------------------------------
4256518 miscalculated calloc() during dlclose/tsorting can result in segv
4254171 DT_SPARC_REGISTER has invalid value associated with it.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-07
        Solaris/SunOS 5.7_x86           patch 106951-07
        Solaris/SunOS 5.6_sparc         patch 107733-04  (never released)
        Solaris/SunOS 5.6_x86           patch 107734-04
--------------------------------------------------------------------------------
4293159 ld needs to combine sections with and without SHF_ORDERED flag(comdat)
4292238 linking a library which has a static char ptr invokes mprotect() call
--------------------------------------------------------------------------------
All the above changes except for:
    4256518 miscalculated calloc() during dlclose/tsorting can result in segv
    4254171 DT_SPARC_REGISTER has invalid value associated with it.
plus:
    4238973 fix for 4121152 affects linking of Ada objects
    4158744 patch 103627-02 causes core when RPATH has blank entry and
            dlopen/dlclose is used
are incorporated in the following patches:
        Solaris/SunOS 5.5.1_sparc       patch 103627-13
        Solaris/SunOS 5.5.1_x86         patch 103628-12
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-08
        Solaris/SunOS 5.7_x86           patch 106951-08
        Solaris/SunOS 5.6_sparc         patch 107733-05
        Solaris/SunOS 5.6_x86           patch 107734-05
--------------------------------------------------------------------------------
4295613 COMMON symbol resolution can be incorrect
--------------------------------------------------------------------------------
All the above changes plus:
    4238973 fix for 4121152 affects linking of Ada objects
    4158744 patch 103627-02 causes core when RPATH has blank entry and
            dlopen/dlclose is used
are incorporated in the following patches:
        Solaris/SunOS 5.5.1_sparc       patch 103627-14
        Solaris/SunOS 5.5.1_x86         patch 103628-13
--------------------------------------------------------------------------------
All the above changes plus:
    4351197 nfs performance problem by 103627-13
are incorporated in the following patches:
        Solaris/SunOS 5.5.1_sparc       patch 103627-15
        Solaris/SunOS 5.5.1_x86         patch 103628-14
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-09
        Solaris/SunOS 5.7_x86           patch 106951-09
        Solaris/SunOS 5.6_sparc         patch 107733-06
        Solaris/SunOS 5.6_x86           patch 107734-06
--------------------------------------------------------------------------------
4158971 increase the default segment alignment for i386 to 64k
4064994 Add an $ISALIST token to those understood by the dynamic linker
xxxxxxx ia64 common code putback
4239308 LD_DEBUG busted for sparc machines
4239008 Support MAP_ANON
4238494 link-auditing extensions required
4232239 R_SPARC_LOX10 truncates field
4231722 R_SPARC_UA* relocations are busted
4235514 R_SPARC_OLO10 relocation fails
4244025 sgsmsg update
4239281 need to support SECREL relocations for ia64
4253751 ia64 linker must support PT_IA_64_UNWIND tables
4259254 dlmopen mistakenly closes fd 0 (stdin) under certain error conditions
4260872 libelf hangs when libthread present
```

```
4224569 linker core dumping when profiling specified
4270937 need mechanism to suppress ld.so.1's use of a default search path.
1050476 ld.so to permit configuration of search path
4273654 filtee processing using $ISALIST could be optimized
4271860 get MERCED cruft out of elf.h
4248991 Dynamic loader (via PLT) corrupts register G4
4275754 cannot mmap file: Resource temporarily unavailable
4277689 The linker can not handle relocation against MOVE tabl
4270766 atexit processing required on dlclose().
4279229 Add a "release" token to those understood by the dynamic linker
4215433 ld can bus error when insufficient disc space exists for output file
4285571 Pssst, want some free disk space?  ld's miscalculating.
4286236 ar gives confusing "bad format" error with a null .stab section
4286838 ld.so.1 can't handle a no-bits segment
4287364 ld.so.1 runtime configuration cleanup
4289573 disable linking of ia64 binaries for Solaris8
4293966 crle(1)'s default directories should be supplied
--------------------------------------------------------------------------------

------------------------------------
Solaris 8 600 (1st Q-update - s28u1)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4309212 dlsym can't find symbol
4311226 rejection of preloading in secure apps is inconsistent
4312449 dlclose: invalid deletion of dependency can occur using RTLD_GLOBAL
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-01
        Solaris/SunOS 5.8_x86           patch 109148-01
        Solaris/SunOS 5.7_sparc         patch 106950-10
        Solaris/SunOS 5.7_x86           patch 106951-10
        Solaris/SunOS 5.6_sparc         patch 107733-07
        Solaris/SunOS 5.6_x86           patch 107734-07
--------------------------------------------------------------------------------

------------------------------------
Solaris 8 900 (2nd Q-update - s28u2)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4324775 non-PIC code & -zcombreloc don't mix very well...
4327653 run-time linker should preload tables it will process (madvise)
4324324 shared object code can be referenced before .init has fired
4321634 .init firing of multiple INITFIRST objects can fail
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-03
        Solaris/SunOS 5.8_x86           patch 109148-03
        Solaris/SunOS 5.7_sparc         patch 106950-11
        Solaris/SunOS 5.7_x86           patch 106951-11
        Solaris/SunOS 5.6_sparc         patch 107733-08
        Solaris/SunOS 5.6_x86           patch 107734-08
--------------------------------------------------------------------------------
4338812 crle(1) omits entries in the directory cache
4341496 RFE: provide a static version of /usr/bin/crle
4340878 rtld should treat $ORIGIN like LD_LIBRARY_PATH in security issues
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-04
        Solaris/SunOS 5.8_x86           patch 109148-04
        Solaris/SunOS 5.7_sparc         patch 106950-12
        Solaris/SunOS 5.7_x86           patch 106951-12
--------------------------------------------------------------------------------
4349563 auxiliary filter error handling regression introduced in 4165487
```

```
4355795 ldd -r now gives "displacement relocated" warnings
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.7_sparc         patch 106950-13
        Solaris/SunOS 5.7_x86           patch 106951-13
        Solaris/SunOS 5.6_sparc         patch 107733-09
        Solaris/SunOS 5.6_x86           patch 107734-09
--------------------------------------------------------------------------------
4210412 versioning a static executable causes ld to core dump
4219652 Linker gives misleading error about not finding main (xarch=v9)
4103449 ld command needs a command line flag to force 64-bits
4187211 problem with RDISP32 linking in copy-relocated objects
4287274 dladdr, dlinfo do not provide the full path name of a shared object
4297563 dlclose still does not remove all objects.
4250694 rtld_db needs a new auxvec entry
4235315 new features for rtld_db (DT_CHECKSUM, dynamic linked .o files
4303609 64bit libelf.so.1 does not properly implement elf_hash()
4310901 su.static fails when OSNet build with lazy-loading
4310324 elf_errno() causes Bus Error(coredump) in 64-bit multithreaded programs
4306415 ld core dump
4316531 BCP: possible failure with dlclose/_preexec_exit_handlers
4313765 LD_BREADTH should be shot
4318162 crle uses automatic strings in putenv.
4255943 Description of -t option incomplete.
4322528 sgs message test infrastucture needs improvement
4239213 Want an API to obtain linker's search path
4324134 use of extern mapfile directives can contribute unused symbols
4322581 ELF data structures could be layed out more efficiently...
4040628 Unnecessary section header symbols should be removed from .dynsym
4300018 rtld: bindlock should be freed before calling call_fini()
4336102 dlclose with non-deletable objects can mishandle dependencies
4329785 mixing of SHT_SUNW_COMDAT & SHF_ORDERED causes ld to seg fault
4334617 COPY relocations should be produces for references to .bss symbols
4248250 relcoation of local ABS symbols incorrect
4335801 For complimentary alignments eliminate ld: warning: symbol 'll'
        has differing a
4336980 ld.so.1 relative path processing revisited
4243097 dlerror(3DL) is not affected by setlocale(3C).
4344528 dump should remove -D and -l usage message
xxxxxxx enable LD_ALTEXEC to access alternate link-editor
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-06
        Solaris/SunOS 5.8_x86           patch 109148-06
--------------------------------------------------------------------------------

------------------------------------
Solaris 8 101 (3rd Q-update - s28u3)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4346144 link-auditing: plt_tracing fails if LA_SYMB_NOPLTENTER given after
        being bound
4346001 The ld should support mapfile syntax to generate PT_SUNWSTACK segment
4349137 rtld_db: A third fallback method for locating the linkmap
4343417 dladdr interface information inadequate
4343801 RFE: crle(1): provide option for updating configuration files
4346615 ld.so.1 attempting to open a directory gives: No such device
4352233 crle should not honor umask
4352330 LD_PRELOAD cannot use absolute path for privileged program
4357805 RFE: man page for ld(1) does not document all -z or -B options in
        Solaris 8 9/00
4358751 ld.so.1: LD_XXX environ variables and LD_FLAGS should be synchronized.
4358862 link editors should reference "64" symlinks instead of sparcv9 (ia64).
4356879 PLTs could use faster code sequences in some cases
4367118 new fast baplt's fail when traversed twice in threaded application
```

```
4366905 Need a way to determine path to a shared library
4351197 nfs performance problem by 103627-13
4367405 LD_LIBRARY_PATH_64 not being used
4354500 SHF_ORDERED ordered scections does not properly sort sections
4369068 ld(1)'s weak symbol processing is inefficient (slow and doesn't scale).
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-07
        Solaris/SunOS 5.8_x86           patch 109148-07
        Solaris/SunOS 5.7_sparc         patch 106950-14
        Solaris/SunOS 5.7_x86           patch 106951-14
--------------------------------------------------------------------------------

------------------------------------
Solaris 8 701 (5th Q-update - s28u5)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4368846 ld(1) fails to version some interfaces given in a mapfile
4077245 dump core dump on null pointer.
4372554 elfdump should demangle symbols (like nm, dump)
4371114 dlclose may unmap a promiscuous object while it's still in use.
4204447 elfdump should understand SHN_AFTER/SHN_BEGIN macro
4377941 initialization of interposers may not occur
4381116 ldd/ld.so.1 could aid in detecting unused dependencies
4381783 dlopen/dlclose of a libCrun+libthread can dump core
4385402 linker & run-time linker must support gABI ELF updates
4394698 ld.so.1 does not process DF_SYMBOLIC - not gABI conforming
4394212 the link editor quietly ignores missing support libraries
4390308 ld.so.1 should provide more flexibility LD_PRELOAD'ing 32-bit/64-bit
        objects
4401232 crle(1) could provide better flexibility for alternatives
4401815 fix misc nits in debugging output...
4402861 cleanup /usr/demo/link_audit & /usr/tmp/librtld_db demo source code...
4393044 elfdump should allow raw dumping of sections
4413168 SHF_ORDERED bit causes linker to generate a separate section
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-08
        Solaris/SunOS 5.8_x86           patch 109148-08
--------------------------------------------------------------------------------
4452202 Typos in <sys/link.h>
4452220 dump doesn't support RUNPATH
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-09
        Solaris/SunOS 5.8_x86           patch 109148-09
--------------------------------------------------------------------------------

-------------------------------------
Solaris 8 1001 (6th Q-update - s28u6)
-------------------------------------
Bugid   Risk Synopsis
================================================================================
4421842 fixups in SHT_GROUP processing required...
4450433 problem with liblddbg output on -Dsection,detail when
        processing SHF_LINK_ORDER
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-10
        Solaris/SunOS 5.8_x86           patch 109148-10
        Solaris/SunOS 5.7_sparc         patch 106950-15
        Solaris/SunOS 5.7_x86           patch 106951-15
--------------------------------------------------------------------------------
4463473 pldd showing wrong output
--------------------------------------------------------------------------------
```

```
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-11
        Solaris/SunOS 5.8_x86           patch 109148-11
--------------------------------------------------------------------------------

-----------------------------------
Solaris 8 202 (7th Q-update - s28u7)
-----------------------------------
Bugid   Risk Synopsis
================================================================================
4488954 ld.so.1 reuses same buffer to send ummapping range to
        _preexec_exit_handlers()
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-12
        Solaris/SunOS 5.8_x86           patch 109148-12
--------------------------------------------------------------------------------

---------
Solaris 9
---------
Bugid   Risk Synopsis
================================================================================
4505289 incorrect handling of _START_ and _END_
4506164 mcs does not recognize #linkbefore or #linkafter qualifiers
4447560 strip is creating unexecutable files...
4513842 library names not in ld.so string pool cause corefile bugs
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-13
        Solaris/SunOS 5.8_x86           patch 109148-13
        Solaris/SunOS 5.7_sparc         patch 106950-16
        Solaris/SunOS 5.7_x86           patch 106951-16
--------------------------------------------------------------------------------
4291384 ld -M with a mapfile does not properly align Fortran REAL*8 data
4413322 SunOS 5.9 librtld_db doesn't show dlopened ".o" files anymore?
4429371 librtld_db busted on ia32 with SC6.x compilers...
4418274 elfdump dumps core on invalid input
4432224 libelf xlate routines are out of date
4433643 Memory leak using dlopen()/dlclose() in Solaris 8
4446564 ldd/lddstub - core dump conditions
4446115 translating SUNW_move sections is broken
4450225 The rdb command can fall into an infinite loop
4448531 Linker Causes Segmentation Fault
4453241 Regression in 4291384 can result in empty symbol table.
4453398 invalid runpath token can cause ld to spin.
4460230 ld (for OS 5.8 and 5.9) loses error message
4462245 ld.so.1 core dumps when executed directly...
4455802 need more flexibility in establishing a support library for ld
4467068 dyn_plt_entsize not properly initialized in ld.so.1
4468779 elf_plt_trace_write() broken on i386 (link-auditing)
4465871 -zld32 and -zld64 does not work the way it should
4461890 bad shared object created with -zredlocsym
4469400 ld.so.1: is_so_loaded isn't as efficient as we thought...
4469566 lazy loading fallback can reference un-relocated objects
4470493 libelf incorectly translates NOTE sections accross architectures...
4469684 rtld leaks dl_handles and permits on dlopen/dlclose
4475174 ld.so.1 prematurly reports the failure to load a object...
4475514 ld.so.1 can core dump in memory allocation fails (no swap)
4481851 Setting ld.so.1 environment variables globally would be useful
4482035 setting LD_PROFILE & LD_AUDIT causes ping command to issue warnings
        on 5.8
4377735 segment reservations cause sbrk() to fail
4491434 ld.so.1 can leak file-descriptors when loading same named objects
4289232 some of warning/error/debugging messages from libld.so can be revised
4462748 Linker Portion of TLS Support
```

```
4496718 run-time linkers mutex_locks not working with ld_libc interface
4497270 The -zredlocsym option should not eliminate partially initialized local
        symbols
4496963 dumping an object with crle(1) that uses $ORIGIN can loose its
        dependencies
4499413 Sun linker orders of magnitude slower than gnu linker
4461760 lazy loading libXm and libXt can fail.
4469031 The partial initialized (local) symbols for intel platform is not
        working.
4492883 Add link-editor option to multi-pass archives to resolve unsatisfied
        symbols
4503731 linker-related commands misspell "argument"
4503768 whocalls(1) should output messages to stderr, not stdout
4503748 whocalls(1) usage message and manpage could be improved
4503625 nm should be taught about TLS symbols - that they aren't allowed that is
4300120 segment address validation is too simplistic to handle segment
        reservations
4404547 krtld/reloc.h could have better error message, has typos
4270931 R_SPARC_HIX22 relocation is not handled properly
4485320 ld needs to support more the 32768 PLTs
4516434 sotruss can not watch libc_psr.so.1
4213100 sotruss could use more flexible pattern matching
4503457 ld seg fault with comdat
4510264 sections with SHF_TLS can come in different orders...
4518079 link-editor support library unable to modify section header flags
4515913 ld.so.1 can incorrectly decrement external reference counts on dlclose()
4519569 ld -V does not return a interesting value...
4524512 ld.so.1 should allow alternate termination signals
4524767 elfdump dies on bogus sh_name fields...
4524735 ld getopt processing of '-' changed
4521931 subroutine in a shared object as LOCL instead of GLOB
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-14
        Solaris/SunOS 5.8_x86           patch 109148-14
        Solaris/SunOS 5.7_sparc         patch 106950-17
        Solaris/SunOS 5.7_x86           patch 106951-17
--------------------------------------------------------------------------------
4532729 tentative definition of TLS variable causes linker to dump core
4526745 fixup ld error message about duplicate dependencies/needed names
4522999 Solaris linker one order of magnitude slower than GNU linker
4518966 dldump undoes existing relocations with no thought of alignment or size.
4587441 Certain libraries have race conditions when setting error codes
4523798 linker option to align bss to large pagesize alignments.
4524008 ld can improperly set st_size of symbols named "_init" or "_fini"
4619282 ld cannot link a program with the option -sb
4620846 Perl Configure probing broken by ld changes
4621122 multiple ld '-zinitarray=' on a commandline fails
--------------------------------------------------------------------------------
        Solaris/SunOS 5.8_sparc         patch 109147-15
        Solaris/SunOS 5.8_x86           patch 109148-15
        Solaris/SunOS 5.7_sparc         patch 106950-18
        Solaris/SunOS 5.7_x86           patch 106951-18
        Solaris/SunOS 5.6_sparc         patch 107733-10
        Solaris/SunOS 5.6_x86           patch 107734-10
--------------------------------------------------------------------------------
All the above changes plus:
        4616944 ar seg faults when order of object file is reversed.
are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch 109147-16
        Solaris/SunOS 5.8_x86           patch 109148-16
--------------------------------------------------------------------------------
All the above changes plus:
        4872634 Large LD_PRELOAD values can cause SEGV of process
are incorporated in the following patches:
        Solaris/SunOS 5.6_sparc         patch T107733-11
        Solaris/SunOS 5.6_x86           patch T107734-11
-------------------------------------------------------------------------------

-----------------------------------
Solaris 9 1202 (2nd Q-update - s9u2)
-----------------------------------
Bugid   Risk Synopsis
===============================================================================
4546416 add help messages to ld.so mdbmodule
4526752 we should build and ship ld.so's mdb module
4624658 update 386 TLS relocation values
4622472 LA_SYMB_DLSYM not set for la_symbind() invocations
4638070 ldd/ld.so.1 could aid in detecting unreferenced dependencies
        PSARC/2002/096 Detecting unreferenced dependencies with ldd(1)
4633860 Optimization for unused static global variables
        PSARC/2002/113 ld -zignore - section elimination
4642829 ld.so.1 mprotect()'s text segment for weak relocations (it shouldn't)
4621479 'make' in $SRC/cmd/sgs/tools tries to install things in the proto area
4529912 purge ia64 source from sgs
4651709 dlopen(RTLD_NOLOAD) can disable lazy loading
4655066 crle: -u with nonexistent config file doesn't work
4654406 string tables created by the link-editor could be smaller...
        PSARC/2002/160 ld -znocompstrtab - disable string-table compression
4651493 RTLD_NOW can result in binding to an object prior to its init being run.
4662575 linker displacement relocation checking introduces significant
        linker overhead
4533195 ld interposes on malloc()/free() preventing support library from freeing
        memory
4630224 crle get's confused about memory layout of objects...
4664855 crle on application failed with ld.so.1 encountering mmap() returning
        ENOMEM err
4669582 latest dynamic linker causes libthread _init to get skipped
4671493 ld.so.1 inconsistantly assigns PATHNAME() on primary objects
4668517 compile with map.bssalign doesn't copy _iob to bss
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-01
        Solaris/SunOS 5.8_sparc         patch T109147-17
        Solaris/SunOS 5.8_x86           patch T109148-17
-------------------------------------------------------------------------------
4701749 On Solaris 8 + 109147-16 ld crashes when building a dynamic library.
4707808 The ldd command is broken in the latest 2.8 linker patch.
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-02
        Solaris/SunOS 5.8_sparc         patch T109147-18
        Solaris/SunOS 5.8_x86           patch T109148-18
-------------------------------------------------------------------------------
4696204 enable extended section indexes in relocatable objects
        PSARC/2001/332 ELF gABI updates - round II
        PSARC/2002/369 libelf interfaces to support ELF Extended Sections
4706503 linkers need to cope with EF_SPARCV9_PSO/EF_SPARCV9_RMO
4716929 updating of local register symbols in dynamic symtab busted...
4710814 add "official" support for the "symbolic" keyword in linker map-file
        PSARC/2002/439 linker mapfile visibility declarations
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-03
        Solaris/SunOS 5.8_sparc         patch T109147-19
        Solaris/SunOS 5.8_x86           patch T109148-19
        Solaris/SunOS 5.7_sparc         patch T106950-19
        Solaris/SunOS 5.7_x86           patch T106951-19
-------------------------------------------------------------------------------

----------------------------------
Solaris 9 403 (3nd Q-update - s9u3)
```

```
----------------------------------
Bugid   Risk Synopsis
===============================================================================
4731174 strip(1) does not fixup SHT_GROUP data
4733697 -zignore with gcc may exclude C++ exception sections
4733317 R_SPARC_*_HIX22 calculations are wrong with 32bit LD building
        ELF64 binaries
4735165 fatal linker error when compiling C++ programs with -xlinkopt
4736951 The mcs broken when the target file is an archive file
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.8_sparc         patch T109147-20
        Solaris/SunOS 5.8_x86           patch T109148-20
        Solaris/SunOS 5.7_sparc         patch T106950-20
        Solaris/SunOS 5.7_x86           patch T106951-20
-------------------------------------------------------------------------------
4739660 Threads deadlock in schedlock and dynamic linker lock.
4653148 ld.so.1/libc should unregister its dlclose() exit handler via a fini.
4743413 ld.so.1 doesn't terminate argv with NULL pointer when invoked directly
4746231 linker core-dumps when SECTION relocations are made against discarded
        sections
4730433 ld.so.1 wastes time repeatedly opening dependencies
4744337 missing RD_CONSISTENT event with dlmopen(LD_ID_NEWLM, ...)
4670835 rd_load_objiter can ignore callback's return value
4745932 strip utility doesn't strip out Dwarf2 debug section
4754751 "strip" command doesn't remove comdat stab sections.
4755674 Patch 109147-18 results in coredump.
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-04
        Solaris/SunOS 5.7_sparc         patch T106950-21
        Solaris/SunOS 5.7_x86           patch T106951-21
-------------------------------------------------------------------------------
4772927 strip core dumps on an archive library
4774727 direct-bindings can fail against copy-reloc symbols
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-05
        Solaris/SunOS 5.9_x86           patch T113986-01
        Solaris/SunOS 5.8_sparc         patch T109147-21
        Solaris/SunOS 5.8_x86           patch T109148-21
        Solaris/SunOS 5.7_sparc         patch T106950-22
        Solaris/SunOS 5.7_x86           patch T106951-22
-------------------------------------------------------------------------------

----------------------------------
Solaris 9 803 (4th Q-update - s9u4)
----------------------------------
Bugid   Risk Synopsis
===============================================================================
4730110 ld.so.1 list implementation could scale better
4728822 restrict the objects dlsym() searches.
        PSARC/2002/478 New dlopen(3dl) flag - RTLD_FIRST
4714146 crle: 64-bit secure pathname is incorrect.
4504895 dlclose() does not remove all objects
4698800 Wrong comments in /usr/lib/ld/sparcv9/map.*
4745129 dldump is inconsistent with .dynamic processing errors.
4753066 LD_SIGNAL isn't very useful in a threaded environment
        PSARC/2002/569 New dlinfo(3dl) flag - RTLD_DI_SIGNAL
4765536 crle: symbolic links can confuse alternative object configuration info
4766815 ld -r of object the TLS data fails
4770484 elfdump can not handle stripped archive file
4770494 The ld command gives improper error message handling broken archive
4775738 overwriting output relocation table when 'ld -zignore' is used
4778247 elfdump -e of core files fails
4779976 elfdump dies on bad relocation entries
```

```
4787579 invalid SHT_GROUP entries can cause linker to seg fault
4783869 dlclose: filter closure exhibits hang/failure - introduced with 4504895
4778418 ld.so.1: there be nits out there
4792461 Thread-Local Storage - x86 instruction sequence updates
        PSARC/2002/746 Thread-Local Storage - x86 instruction sequence updates
4461340 sgs: ugly build output while suppressing ia64 (64-bit) build on Intel
4790194 dlopen(..., RTLD_GROUP) has an odd interaction with interposition
4804328 auditing of threaded applications results in deadlock
4806476 building relocatable objects with SHF_EXCLUDE loses relocation
        information
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-06
        Solaris/SunOS 5.9_x86           patch T113986-02
        Solaris/SunOS 5.8_sparc         patch T109147-22
        Solaris/SunOS 5.8_x86           patch T109148-22
--------------------------------------------------------------------------------
4731183 compiler creates .tlsbss section instead of .tbss as documented
4816378 TLS: a tls test case dumps core with C and C++ compilers
4817314 TLS_GD relocations against local symbols do not reference symbol...
4811951 non-default symbol visibility overriden by definition in shared object
4802194 relocation error of mozilla built by K2 compiler
4715815 ld should allow linking with no output file (or /dev/null)
4793721 Need a way to null all code in ISV objects enabling ld performance
        tuning
--------------------------------------------------------------------------------
All the above changes plus:
        4796237 RFE: link-editor became extremely slow with patch 109147-20 and
                static libraries
are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-07
        Solaris/SunOS 5.9_x86           patch T113986-03
        Solaris/SunOS 5.8_sparc         patch T109147-23
        Solaris/SunOS 5.8_x86           patch T109148-23
--------------------------------------------------------------------------------

------------------------------------
Solaris 9 1203 (5th Q-update - s9u5)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4830584 mmap for the padding region doesn't get freed after dlclose
4831650 ld.so.1 can walk off the end of it's call_init() array...
4831544 ldd using .so modules compiled with FD7 compiler caused a core dump
4834784 Accessing members in a TLS structure causes a core dump in Oracle
4824026 segv when -z combreloc is used with -xlinkopt
4825296 typo in elfdump
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-08
        Solaris/SunOS 5.9_x86           patch T113986-04
        Solaris/SunOS 5.8_sparc         patch T109147-24
        Solaris/SunOS 5.8_x86           patch T109148-24
--------------------------------------------------------------------------------
4470917 Solaris Process Model Unification (link-editor components only)
        PSARC/2002/117 Solaris Process Model Unification
4744411 Bloomberg wants a faster linker.
4811969 64-bit links can be much slower than 32-bit.
4825065 ld(1) should ignore consecutive empty sections.
4838226 unrelocated shared objects may be erroneously collected for init firing
4830889 TLS: testcase coredumps with -xarch=v9 and -g
4845764 filter removal can leave dangling filtee pointer
4811093 apptrace -F libc date core dumps
4826315 Link editors need to be pre- and post- Unified Process Model aware
4868300 interposing on direct bindings can fail
4872634 Large LD_PRELOAD values can cause SEGV of process
```

```
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-09
        Solaris/SunOS 5.9_x86           patch T113986-05
        Solaris/SunOS 5.8_sparc         patch T109147-25
        Solaris/SunOS 5.8_x86           patch T109148-25
--------------------------------------------------------------------------------

------------------------------------
Solaris 9 404 (6th Q-update - s9u6)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4870260 The elfdump command should produce more warning message on invalid move
        entries.
4865418 empty PT_TLS program headers cause problems in TLS enabled applications
4825151 compiler core dumped with a -mt -xF=%all test
4845829 The runtime linker fails to dlopen() long path name.
4900684 shared libraries with more then 32768 plt's fail for sparc ELF64
4906062 Makefiles under usr/src/cmd/sgs needs to be updated
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-10
        Solaris/SunOS 5.9_x86           patch T113986-06
        Solaris/SunOS 5.8_sparc         patch T109147-26
        Solaris/SunOS 5.8_x86           patch T109148-26
        Solaris/SunOS 5.7_sparc         patch T106950-24
        Solaris/SunOS 5.7_x86           patch T106951-24
--------------------------------------------------------------------------------
4900320 rtld library mapping could be faster
4911775 implement GOTDATA proposal in ld
        PSARC/2003/477 SPARC GOTDATA instruction sequences
4904565 Functionality to ignore relocations against external symbols
4764817 add section types SHT_DEBUG and SHT_DEBUGSTR
        PSARC/2003/510 New ELF DEBUG and ANNOTATE sections
4850703 enable per-symbol direct bindings
4716275 Help required in the link analysis of runtime interfaces
        PSARC/2003/519 Link-editors: Direct Binding Updates
4904573 elfdump may hang when processing archive files
4918310 direct binding from an executable can't be interposed on
4918938 ld.so.1 has become SPARC32PLUS - breaks 4.x binary compatibility
4911796 S1S8 C++: ld dump core when compiled and linked with xlinkopt=1.
4889914 ld crashes with SEGV using -M mapfile under certain conditions
4911936 exception are not catch from shared library with -zignore
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-11
        Solaris/SunOS 5.9_x86           patch T113986-07
        Solaris/SunOS 5.8_sparc         patch T109147-27
        Solaris/SunOS 5.8_x86           patch T109148-27
        Solaris/SunOS 5.7_sparc         patch T106950-25
        Solaris/SunOS 5.7_x86           patch T106951-25
--------------------------------------------------------------------------------
4946992 ld crashes due to huge number of sections (>65,000)
4951840 mcs -c goes into a loop on executable program
4939869 Need additional relocation types for abs34 code model
        PSARC/2003/684 abs34 ELF relocations
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-12
        Solaris/SunOS 5.9_x86           patch T113986-08
        Solaris/SunOS 5.8_sparc         patch T109147-28
        Solaris/SunOS 5.8_x86           patch T109148-28
--------------------------------------------------------------------------------

------------------------------------
```

```
Solaris 9 904 (7th Q-update - s9u7)
------------------------------------
Bugid   Risk Synopsis
================================================================================
4912214 Having multiple of libc.so.1 in a link map causes malloc() to fail
4526878 ld.so.1 should pass MAP_ALIGN flag to give kernel more flexibility
4930997 sgs bld_vernote.ksh script needs to be hardend...
4796286 ld.so.1: scenario for trouble?
4930985 clean up cruft under usr/src/cmd/sgs/tools
4933300 remove references to Ultra-1 in librtld_db demo
4936305 string table compression is much too slow...
4939626 SUNWonld internal package must be updated...
4939565 per-symbol filtering required
4948119 ld(1) -z loadfltr fails with per-symbol filtering
4948427 ld.so.1 gives fatal error when multiple RTLDINFO objects are loaded
4940894 ld core dumps using "-xldscope=symbolic
4955373 per-symbol filtering refinements
4878827 crle(1M) - display post-UPM search paths, and compensate for pre-UPM.
4955802 /usr/ccs/bin/ld dumps core in process_reld()
4964415 elfdump issues wrong relocation error message
4966465 LD_NOAUXFLTR fails when object is both a standard and auxiliary filter
4973865 the link-editor does not scale properly when linking objects with
        lots of syms
4975598 SHT_SUNW_ANNOTATE section relocation not resolved
4974828 nss_files nss_compat r_mt tests randomly segfaulting
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-13
        Solaris/SunOS 5.9_x86           patch T113986-09
--------------------------------------------------------------------------------
4860508 link-editors should create/promote/verify hardware capabilities
5002160 crle: reservation for dumped objects gets confused by mmaped object
4967869 linking stripped library causes segv in linker
5006657 link-editor doesn't always handle nodirect binding syminfo information
4915901 no way to see ELF information
5021773 ld.so.1 has trouble with objects having more than 2 segments.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-14
        Solaris/SunOS 5.9_x86           patch T113986-10
        Solaris/SunOS 5.8_sparc         patch T109147-29
        Solaris/SunOS 5.8_x86           patch T109148-29
--------------------------------------------------------------------------------
All the above changes plus:
        6850124 dlopen reports "No such file or directory" in spite of ENOMEM
                when mmap fails in anon_map()
are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch TXXXXXX-XX
        Solaris/SunOS 5.9_x86           patch TXXXXXX-XX
--------------------------------------------------------------------------------

----------
Solaris 10
----------
Bugid   Risk Synopsis
================================================================================
5044797 ld.so.1: secure directory testing is being skipped during filtee
        processing
4963676 Remove remaining static libraries
5021541 unnecessary PT_SUNWBSS segment may be created
5031495 elfdump complains about bad symbol entries in core files
5012172 Need error when creating shared object with .o compiled
        -xarch=v9 -xcode=abs44
4994738 rd_plt_resolution() resolves ebx-relative PLT entries incorrectly
5023493 ld -m output with patch 109147-25 missing .o information
--------------------------------------------------------------------------------
```

```
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-15
        Solaris/SunOS 5.9_x86           patch T113986-11
        Solaris/SunOS 5.8_sparc         patch T109147-30
        Solaris/SunOS 5.8_x86           patch T109148-30
--------------------------------------------------------------------------------
5071614 109147-29 & -30 break the build of on28-patch on Solaris 8 2/04
5029830 crle: provide for optional alternative dependencies.
5034652 ld.so.1 should save, and print, more error messages
5036561 ld.so.1 outputs non-fatal fatal message about auxiliary filter libraries
5042713 4866170 broke ld.so's ::setenv
5047082 ld can core dump on bad gcc objects
5047612 ld.so.1: secure pathname verification is flawed with filter use
5047235 elfdump can core dump printing PT_INTERP section
4798376 nits in demo code
5041446 gelf_update_*() functions inconsistently return NULL or 0
5032364 M_ID_TLSBSS and M_ID_UNKNOWN have the same value
4707030 Empty LD_PRELOAD_64 doesn't override LD_PRELOAD
4968618 symbolic linkage causes core dump
5062313 dladdr() can cause deadlock in MT apps.
5056867 $ISALIST/$HWCAP expansion should be more flexible.
4918303 0@0.so.1 should not use compiler-supplied crt*.o files
5058415 whocalls cannot take more than 10 arguments
5067518 The fix for 4918303 breaks the build if a new work space is used.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-16
        Solaris/SunOS 5.9_x86           patch T113986-12
        Solaris/SunOS 5.8_sparc         patch T109147-31
        Solaris/SunOS 5.8_x86           patch T109148-31
--------------------------------------------------------------------------------
5013759 *file* should report hardware/software capabilities (link-editor
        components only)
5063580 libldstab: file /tmp/posto..: .stab[.index|.sbfocus] found with no
        matching stri
5076838 elfdump(1) is built with a CTF section (the wrong one)
5080344 Hardware capabilities are not enforced for a.out
5079061 RTLD_DEFAULT can be expensive
        PSARC/2004/747 New dlsym(3c) Handle - RTLD_PROBE
5064973 allow normal relocs against TLS symbols for some sections
5085792 LD_XXXX_64 should override LD_XXXX
5096272 every executable or library has a .SUNW_dof section
5094135 Bloomberg wants a faster ldd.
5086352 libld.so.3 should be built with a .SUNW_ctf ELF section, ready for CR
5098205 elfdump gives wrong section name for the global offset table
5092414 Linker patch 109147-29 makes Broadvison One-To-One server v4.1
        installation fail
5080256 dump(1) doesn't list ELF hardware capabilities
5097347 recursive read lock in gelf_getsym()
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-17
        Solaris/SunOS 5.9_x86           patch T113986-13
        Solaris/SunOS 5.8_sparc         patch T109147-32
        Solaris/SunOS 5.8_x86           patch T109148-32
--------------------------------------------------------------------------------
5106206 ld.so.1 fail to run a Solaris9 program that has libc linked with
        -z lazyload
5102601 ON should deliver a 64-bit operating system for Opteron systems
        (link-editor components only)
6173852 enable link_auditing technology for amd64
6174599 linker does not create .eh_frame_hdr sections for eh_frame sections
        with SHF_LINK_ORDER
6175609 amd64 run-time linker has a corrupted note section
6175843 amd64 rdb_demo files not installed
6182293 ld.so.1 can repeatedly relocate object .plts (RTLD_NOW).
```

```
6183645 ld core dumps when automounter fails
6178667 ldd list unexpected (file not found) in x86 environment.
6181928 Need new reloc types R_AMD64_GOTOFF64 and R_AMD64_GOTPC32
6182884 AMD64: ld coredumps when building a shared library
6173559 The ld may set incorrect value for sh_addralign under some conditions.
5105601 ld.so.1 gets a little too enthusiastic with interposition
6189384 ld.so.1 should accommodate a files dev/inode change (libc loopback mnt)
6177838 AMD64: linker cannot resolve PLT for 32-bit a.out(s) on amd64-S2 kernel
6190863 sparc disassembly code should be removed from rdb_demo
6191488 unwind eh_frame_hdr needs corrected encoding value
6192490 moe(1) returns /lib/libc.so.1 for optimal expansion of libc HWCAP
        libraries
6192164 AMD64: introduce dlamd64getunwind interface
        PSARC/2004/747 libc::dlamd64getunwind()
6195030 libdl has bad version name
6195521 64-bit moe(1) missed the train
6198358 AMD64: bad eh_frame_hdr data when C and C++ mixed in a.out
6204123 ld.so.1: symbol lookup fails even after lazy loading fallback
6207495 UNIX98/UNIX03 vsx namespace violation DYNL.hdr/misc/dlfcn/T.dlfcn
        14 Failed
6217285 ctfmerge crashed during full onnv build
--------------------------------------------------------------------------------

-------------------------------------
Solaris 10 106 (1st Q-update - s10u1)
-------------------------------------
Bugid   Risk Synopsis
================================================================================
6209350 Do not include signature section from dynamic dependency library into
        relocatable object
6212797 The binary compiled on SunOS4.x doesn't run on Solaris8 with Patch
        109147-31
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-18
        Solaris/SunOS 5.9_x86           patch T113986-14
        Solaris/SunOS 5.8_sparc         patch T109147-33
        Solaris/SunOS 5.8_x86           patch T109148-33
--------------------------------------------------------------------------------
6219538 112963-17: linker patch causes binary to dump core
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-01
        Solaris/SunOS 5.10_x86          patch T118345-01
        Solaris/SunOS 5.9_sparc         patch T112963-19
        Solaris/SunOS 5.9_x86           patch T113986-15
        Solaris/SunOS 5.8_sparc         patch T109147-34
        Solaris/SunOS 5.8_x86           patch T109148-34
--------------------------------------------------------------------------------
6257177 incremental builds of usr/src/cmd/sgs can fail...
6219651 AMD64: Linker does not issue error for out of range R_AMD64_PC32
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-02
        Solaris/SunOS 5.10_x86          patch T118345-02
        Solaris/SunOS 5.9_sparc         patch T112963-20
        Solaris/SunOS 5.9_x86           patch T113986-16
        Solaris/SunOS 5.8_sparc         patch T109147-35
        Solaris/SunOS 5.8_x86           patch T109148-35
NOTE: The fix for 6219651 is only applicable for 5.10_x86 platform.
--------------------------------------------------------------------------------
5080443 lazy loading failure doesn't clean up after itself (D)
6226206 ld.so.1 failure when processing single segment hwcap filtee
6228472 ld.so.1: link-map control list stacking can loose objects
6235000 random packages not getting installed in snv_09 and snv_10 -
        rtld/common/malloc.c Assertion
```

```
6219317 Large page support is needed for mapping executables, libraries and
        files (link-editor components only)
6244897 ld.so.1 can't run apps from commandline
6251798 moe(1) returns an internal assertion failure message in some
        circumstances
6251722 ld fails silently with exit 1 status when -z ignore passed
6254364 ld won't build libgenunix.so with absolute relocations
6215444 ld.so.1 caches "not there" lazy libraries, foils svc.startd(1M)'s logic
6222525 dlsym(3C) trusts caller(), which may return wrong results with tail call
        optimization
6241995 warnings in sgs should be fixed (link-editor components only)
6258834 direct binding availability should be verified at runtime
6260361 lari shouldn't count a.out non-zero undefined entries as interesting
6260780 ldd doesn't recognize LD_NOAUXFLTR
6266261 Add ld(1) -Bnodirect support (D)
6261990 invalid e_flags error could be a little more friendly
6261803 lari(1) should find more events uninteresting (D)
6267352 libld_malloc provides inadequate alignment
6268693 SHN_SUNW_IGNORE symbols should be allowed to be mulitiply defined
6262789 Infosys wants a faster linker
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-03
        Solaris/SunOS 5.10_x86          patch T118345-03
        Solaris/SunOS 5.9_sparc         patch T112963-21
        Solaris/SunOS 5.9_x86           patch T113986-17
        Solaris/SunOS 5.8_sparc         patch T109147-36
        Solaris/SunOS 5.8_x86           patch T109148-36
--------------------------------------------------------------------------------
6283601 The usr/src/cmd/sgs/packages/common/copyright contains old information
        legally problematic
6276905 dlinfo gives inconsistent results (relative vs absolute linkname) (D)
        PSARC/2005/357 dlinfo(3c) RTLD_DI_ARGSINFO
6284941 excessive link times with many groups/sections
6280467 dlclose() unmaps shared library before library's _fini() has finished
6291547 ld.so mishandles LD_AUDIT causing security problems.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-04
        Solaris/SunOS 5.10_x86          patch T118345-04
        Solaris/SunOS 5.9_sparc         patch T112963-22
        Solaris/SunOS 5.9_x86           patch T113986-18
        Solaris/SunOS 5.8_sparc         patch T109147-37
        Solaris/SunOS 5.8_x86           patch T109148-37
--------------------------------------------------------------------------------
6295971 UNIX98/UNIX03 *vsx* DYNL.hdr/misc/dlfcn/T.dlfcn 14 fails, auxv.h syntax
        error
6299525 .init order failure when processing cycles
6273855 gcc and sgs/crle don't get along
6273864 gcc and sgs/libld don't get along
6273875 gcc and sgs/rtld don't get along
6272563 gcc and amd64/krtld/doreloc.c don't get along
6290157 gcc and sgs/librtld_db/rdb_demo don't get along
6301218 Matlab dumps core on startup when running on 112963-22 (D)
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-06
        Solaris/SunOS 5.10_x86          patch T118345-08
        Solaris/SunOS 5.9_sparc         patch T112963-23
        Solaris/SunOS 5.9_x86           patch T113986-19
        Solaris/SunOS 5.8_sparc         patch T109147-38
        Solaris/SunOS 5.8_x86           patch T109148-38
--------------------------------------------------------------------------------
6314115 Checkpoint refuses to start, crashes on start, after application of
        linker patch 112963-22
--------------------------------------------------------------------------------
```

```
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-24
        Solaris/SunOS 5.9_x86           patch T113986-20
        Solaris/SunOS 5.8_sparc         patch T109147-39
        Solaris/SunOS 5.8_x86           patch T109148-39
-------------------------------------------------------------------------------
6318306 a dlsym() from a filter should be redirected to an associated filtee
6318401 mis-aligned TLS variable
6324019 ld.so.1: malloc alignment is insufficient for new compilers
6324589 psh coredumps on x86 machines on snv_23
6236594 AMD64: Linker needs to handle the new .lbss section (D)
        PSARC 2005/514 AMD64 - large section support
6314743 Linker: incorrect resolution for R_AMD64_GOTPC32
6311865 Linker: x86 medium model; invalid ELF program header
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-07
        Solaris/SunOS 5.10_x86          patch T118345-12
-------------------------------------------------------------------------------
6309061 link_audit should use __asm__ with gcc
6310736 gcc and sgs/libld don't get along on SPARC
6329796 Memory leak with iconv_open/iconv_close with patch 109147-33
6332983 s9 linker patches 112963-24/113986-20 causing cluster machines not
        to boot
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T117461-08
        Solaris/SunOS 5.10_x86          patch T121208-02
        Solaris/SunOS 5.9_sparc         patch T112963-25
        Solaris/SunOS 5.9_x86           patch T113986-21
        Solaris/SunOS 5.8_sparc         patch T109147-40
        Solaris/SunOS 5.8_x86           patch T109148-40
-------------------------------------------------------------------------------
6445311 The sparc S8/S9/S10 linker patches which include the fix for the
        CR6222525 are hit by the CR6439613.
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-26
        Solaris/SunOS 5.8_sparc         patch T109147-41
-------------------------------------------------------------------------------

------------------------------------
Solaris 10 807 (4th Q-update - s10u4)
------------------------------------
Bugid   Risk Synopsis
===============================================================================
6487273 ld.so.1 may open arbitrary locale files when relative path is built
        from locale environment vars
6487284 ld.so.1: buffer overflow in doprf() function
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T124922-01
        Solaris/SunOS 5.10_x86          patch T124923-01
        Solaris/SunOS 5.9_sparc         patch T112963-27
        Solaris/SunOS 5.9_x86           patch T113986-22
        Solaris/SunOS 5.8_sparc         patch T109147-42
        Solaris/SunOS 5.8_x86           patch T109148-41
-------------------------------------------------------------------------------
6477132 ld.so.1: memory leak when running set*id application
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T124922-02
        Solaris/SunOS 5.10_x86          patch T124923-02
        Solaris/SunOS 5.9_sparc         patch T112963-30
        Solaris/SunOS 5.9_x86           patch T113986-24
-------------------------------------------------------------------------------
```

```
6340814 ld.so.1 core dump with HWCAP relocatable object + updated statistics
6307274 crle bug with LD_LIBRARY_PATH
6317969 elfheader limited to 65535 segments (link-editor components only)
6350027 ld.so.1 aborts with assertion failed on amd64
6362044 ld(1) inconsistencies with LD_DEBUG=-Dunused and -zignore
6362047 ld.so.1 dumps core when combining HWCAP and LD_PROFILE
6304206 runtime linker may respect LANG and LC_MESSAGE more than LC_ALL
6363495 Catchup required with Intel relocations
6326497 ld.so not properly processing LD_LIBRARY_PATH ending in :
6307146 mcs dumps core when appending null string to comment section
6371877 LD_PROFILE_64 with gprof does not produce correct results on amd64
6372082 ld -r erroneously creates .got section on i386
6201866 amd64: linker symbol elimination is broken
6372620 printstack() segfaults when called from static function (D)
6380470 32-bit ld(1) incorrectly builds 64-bit relocatable objects
6391407 Insufficient alignment of 32-bit object in archive makes ld segfault
        (libelf component only) (D)
6316708 LD_DEBUG should provide a means of identifying/isolating individual
        link-map lists (P)
6280209 elfdump cores on memory model 0x3
6197234 elfdump and dump don't handle 64-bit symbols correctly
6398893 Extended section processing needs some work
6397256 ldd dumps core in elf_fix_name
6327926 ld does not set etext symbol correctly for AMD64 medium model (D)
6390410 64-bit LD_PROFILE can fail: relocation error when binding profile plt
6382945 AMD64-GCC: dbx: internal error: dwarf reference attribute out of bounds
6262333 init section of .so dlopened from audit interface not being called
6409613 elf_outsync() should fsync()
6426048 C++ exceptions broken in Nevada for amd64
6429418 ld.so.1: need work-around for Nvidia drivers use of static TLS
6429504 crle(1) shows wrong defaults for non-existent 64-bit config file
6431835 data corruption on x64 in 64-bit mode while LD_PROFILE is in effect
6423051 static TLS support within the link-editors needs a major face lift (D)
6388946 attempting to dlopen a .o file mislabeled as .so fails
6446740 allow mapfile symbol definitions to create backing storage (D)
4986360 linker crash on exec of .so (as opposed to a.out) -- error preferred
        instead
6229145 ld: initarray/finiarray processing occurs after got size is determined
6324924 the linker should warn if there's a .init section but not _init
6424132 elfdump inserts extra whitespace in bitmap value display
6449485 ld(1) creates misaligned TLS in binary compiled with -xpg
6424550 Write to unallocated (wua) errors when libraries are built with
        -z lazyload
6464235 executing the 64-bit ld(1) should be easy (D)
6465623 need a way of building unix without an interpreter
6467925 ld: section deletion (-z ignore) requires improvement
6357230 specfiles should be nuked (link-editor components only)
-------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T124922-03
        Solaris/SunOS 5.10_x86          patch T124923-03

These patches also include the framework changes for the following bug fixes.
However, the associated feature has not been enabled in Solaris 10 or earlier
releases:

6174390 crle configuration files are inconsistent across platforms (D, P)
6432984 ld(1) output file removal - change default behavior (D)
        PSARC/2006/353 ld(1) output file removal - change default behavior
-------------------------------------------------------------------------------

------------------------------------
Solaris 10 508 (5th Q-update - s10u5)
------------------------------------
Bugid   Risk Synopsis
===============================================================================
6561987 data vac_conflict faults on lipthread libthread libs in s10.
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T127111-01
        Solaris/SunOS 5.10_x86          patch T127112-01
--------------------------------------------------------------------------------
6501793 GOTOP relocation transition (optimization) fails with offsets > 2^32
6532924 AMD64: Solaris 5.11 55b: SEGV after whocatches
6551627 OGL: SIGSEGV when trying to use OpenGL pipeline with splash screen,
        Solaris/Nvidia only
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T127111-04
        Solaris/SunOS 5.10_x86          patch T127112-04
--------------------------------------------------------------------------------
6479848 Enhancements to the linker support interface needed. (D)
        PSARC/2006/595 link-editor support library interface - ld_open()
6521608 assertion failure in runtime linker related to auditing
6494228 pclose() error when an audit library calls popen() and the main target
        is being run under ldd (D)
6568745 segfault when using LD_DEBUG with bit_audit library when instrumenting
        mozilla (D)
        PSARC/2007/413 Add -zglobalaudit option to ld
6602294 ps_pbrandname breaks apps linked directly against librtld_db
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T127111-07
        Solaris/SunOS 5.10_x86          patch T127112-07
--------------------------------------------------------------------------------

-------------------------------------
Solaris 10 908 (6th Q-update - s10u6)
-------------------------------------
Bugid   Risk Synopsis
================================================================================
6672544 elf_rtbndr must support non-ABI aligned stacks on amd64
6668050 First trip through PLT does not preserve args in xmm registers
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patch:
        Solaris/SunOS 5.10_x86          patch T137138-01
--------------------------------------------------------------------------------

-------------------------------------
Solaris 10 409 (7th Q-update - s10u7)
-------------------------------------
Bugid   Risk Synopsis
================================================================================
6629404 ld with -z ignore doesn’t scale
6606203 link editor ought to allow creation of >2gb sized objects (P)
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T139574-01
        Solaris/SunOS 5.10_x86          patch T139575-01
--------------------------------------------------------------------------------
6746674 setuid applications do not find libraries any more because trusted
        directories behavior changed (D)
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T139574-02
        Solaris/SunOS 5.10_x86          patch T139575-02
--------------------------------------------------------------------------------
6703683 Can’t build VirtualBox on Build 88 or 89
6737579 process_req_lib() in libld consumes file descriptors
6685125 ld/elfdump do not handle ZERO terminator .eh_frame amd64 unwind entry
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
```

```
        Solaris/SunOS 5.10_sparc        patch T139574-03
        Solaris/SunOS 5.10_x86          patch T139575-03
--------------------------------------------------------------------------------

-------------------------------------
Solaris 10 1009 (8th Q-update - s10u8)
-------------------------------------
Bugid   Risk Synopsis
================================================================================
6782597 32-bit ld.so.1 needs to accept objects with large inode number
6805502 The addition of "inline" keywords to sgs code broke the lint
        verification in S10
6807864 ld.so.1 is susceptible to a fatal dlsym()/setlocale() race
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T141692-01
        Solaris/SunOS 5.10_x86          patch T141693-01
NOTE: The fix for 6805502 is only applicable to s10.
--------------------------------------------------------------------------------
6826410 ld needs to sort sections using 32-bit sort keys
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T141771-01
        Solaris/SunOS 5.10_x86          patch T141772-01
NOTE: The fix for 6826410 is also available for s9 in the following patches:
        Solaris/SunOS 5.9_sparc         patch T112963-33
        Solaris/SunOS 5.9_x86           patch T113986-27
--------------------------------------------------------------------------------
6568447 bcp is broken by 6551627
6599700 librtld_db needs better plugin support
6713830 mdb dumped core reading a gcore
6756048 rd_loadobj_iter() should always invoke brand plugin callback
6786744 32-bit dbx failed with unknown rtld_db.so error on snv_104
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T141444-06
        Solaris/SunOS 5.10_x86          patch T141445-06
--------------------------------------------------------------------------------

--------------------------------------
Solaris 10 1005 (9th Q-update - s10u9)
--------------------------------------
Bugid   Risk Synopsis
================================================================================
6850124 dlopen reports "No such file or directory" in spite of ENOMEM
        when mmap fails in anon_map()
6826513 ldd gets confused by a crle(1) LD_PRELOAD setting
6684577 ld should propagate SHF_LINK_ORDER flag to ET_REL objects
6524709 executables using /usr/lib/libc.so.1 as the ELF interpreter dump core
        (link-editor components only)
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch T143895-01
        Solaris/SunOS 5.10_x86          patch T143896-01
--------------------------------------------------------------------------------

----------------------------------------
Solaris 10 XXXX (10th Q-update - s10u10)
----------------------------------------
Bugid   Risk Synopsis
================================================================================
6478684 isainfo/cpuid reports pause instruction not supported on amd64
        PSARC/2010/089 Removal of AV_386_PAUSE and AV_386_MON
--------------------------------------------------------------------------------
All the above changes are incorporated in the following patches:
        Solaris/SunOS 5.10_sparc        patch TXXXXXX-XX
        Solaris/SunOS 5.10_x86          patch TXXXXXX-XX
--------------------------------------------------------------------------------

--------------------------------------------
Solaris Nevada (OpenSolaris 2008.05, snv_86)
--------------------------------------------
Bugid   Risk Synopsis
================================================================================
6409350 BrandZ project integration into Solaris (link-editor components only)
6459189 UNIX03: *VSC* c99 compiler overwrites non-writable file
6423746 add an option to relax the resolution of COMDAT relocs (D)
4934427 runtime linker should load up static symbol names visible to
        dladdr() (D)
        PSARC 2006/526 SHT_SUNW_LDYNSYM - default local symbol addition
6448719 sys/elf.h could be updated with additional machine and ABI types
6336605 link-editors need to support R_*_SIZE relocations
        PSARC/2006/558 R_*_SIZE relocation support
6475375 symbol search optimization to reduce rescans
6475497 elfdump(1) is misreporting sh_link
6482058 lari(1) could be faster, and handle per-symbol filters better
6482974 defining virtual address of text segment can result in an invalid data
        segment
6476734 crle(1m) "-l" as described fails system, crle cores trying to fix
        /a/var/ld/ld.config in failsafe
6487499 link_audit "make clobber" creates and populates proto area
6488141 ld(1) should detect attempt to reference 0-length .bss section
6496718 restricted visibility symbol references should trigger archive
        extraction
6515970 HWCAP processing doesn't clean up fmap structure - browser fails to
        run java applet
6494214 Refinements to symbolic binding, symbol declarations and
        interposition (D)
        PSARC/2006/714 ld(1) mapfile: symbol interpose definition
6475344 DTrace needs ELF function and data symbols sorted by address (D)
        PSARC/2007/026 ELF symbol sort sections
6518480 ld -melf_i386 doesn't complain (D)
6519951 bfu is just another word for exit today (RPATH -> RUNPATH conversion
        bites us) (D)
6521504 ld: hardware capabilities processing from relocatables objects needs
        hardening.
6518322 Some ELF utilities need updating for .SUNW_ldynsym section (D)
        PSARC/2007/074 -L option for nm(1) to display SHT_SUNW_LDYNSYM symbols
6523787 dlopen() handle gets mistakenly orphaned - results in access to freed
        memory
6531189 SEGV in dladdr()
6527318 dlopen(name, RTLD_NOLOAD) returns handle for unloaded library
6518359 extern mapfiles references to _init/_fini can create INIT/FINI
        addresses of 0
6533587 ld.so.1: init/fini processing needs to compensate for interposer
        expectations
6516118 Reserved space needed in ELF dynamic section and string table (D)
        PSARC/2007/127 Reserved space for editing ELF dynamic sections
6535688 elfdump could be more robust in the face of Purify (D)
6516665 The link-editors should be more resilient against gcc's symbol
        versioning
6541004 hwcap filter processing can leak memory
5108874 elfdump SEGVs on bad object file
6547441 Uninitialized variable causes ld.so.1 to crash on object cleanup
6341667 elfdump should check alignments of ELF header elements
6387860 elfdump cores, when processing linux built ELF file
6198202 mcs -d dumps core
6246083 elfdump should allow section index specification
        (numeric -N equivalent) (D)
        PSARC/2007/247 Add -I option to elfdump
6556563 elfdump section overlap checking is too slow for large files
5006034 need ?E mapfile feature extension (D)
```

```
6565476 rtld symbol version check prevents GNU ld binary from running
6567670 ld(1) symbol size/section size verification uncovers Haskell
        compiler inconsistency
6530249 elfdump should handle ELF files with no section header table (D)
        PSARC/2007/395 Add -P option to elfdump
6573641 ld.so.1 does not maintain parent relationship to a dlopen() caller.
6577462 Additional improvements needed to handling of gcc's symbol versioning
6583742 ELF string conversion library needs to lose static writable buffers
6589819 ld generated reference to __tls_get_addr() fails when resolving to a
        shared object reference
6595139 various applications should export yy* global variables for libl
        PSARC/2007/474 new ldd(1) -w option
6597841 gelf_getdyn() reads one too many dynamic entries
6603313 dlclose() can fail to unload objects after fix for 6573641
6234471 need a way to edit ELF objects (D)
        PSARC/2007/509 elfedit
5035454 mixing -Kpic and -KPIC may cause SIGSEGV with -xarch=v9
6473571 strip and mcs get confused and corrupt files when passed
        non-ELF arguments
6253589 mcs has problems handling multiple SHT_NOTE sections
6610591 do_reloc() should not require unused arguments
6602451 new symbol visibilities required: EXPORTED, SINGLETON and ELIMINATE (D)
        PSARC/2007/559 new symbol visibilities - EXPORTED, SINGLETON, and
        ELIMINATE
6570616 elfdump should display incorrectly aligned note section
6614968 elfedit needs string table module (D)
6620533 HWCAP filtering can leave uninitialized data behind - results in
        "rejected: Invalid argument"
6617855 nodirect tag can be ignored when other syminfo tags are available
        (link-editor components only)
6621066 Reduce need for new elfdump options with every section type (D)
        PSARC/2007/620 elfdump -T, and simplified matching
6627765 soffice failure after integration of 6603313 - dangling GROUP pointer.
6319025 SUNWbtool packaging issues in Nevada and S10u1.
6626135 elfedit capabilities str->value mapping should come from
        usr/src/common/elfcap
6642769 ld(1) -z combreloc should become default behavior (D)
        PSARC/2008/006 make ld(1) -z combreloc become default behavior
6634436 XFFLAG should be updated.  (link-editor components only)
6492726 Merge SHF_MERGE|SHF_STRINGS input sections (D)
4947191 OSNet should use direct bindings  (link-editor components only)
6654381 lazy loading fall-back needs optimizing
6658385 ld core dumps when building Xorg on nv_82
6516808 ld.so.1's token expansion provides no escape for platforms that don't
        report HWCAP
6668534 Direct bindings can compromise function address comparisons from
        executables
6667661 Direct bindings can compromise executables with insufficient copy
        relocation information
6357282 ldd should recognize PARENT and EXTERN symbols (D)
        PSARC/2008/148 new ldd(1) -p option
6672394 ldd(1) unused dependency processing is tricked by relocations errors
--------------------------------------------------------------------------------

---------------------------------------------
Solaris Nevada (OpenSolaris 2008.11, snv_101)
---------------------------------------------
Bugid   Risk Synopsis
================================================================================
6671255 link-editor should support cross linking (D)
        PSARC/2008/179 cross link-editor
6674666 elfedit dyn:posflag1 needs option to locate element via NEEDED item
6675591 elfwrap - wrap data in an ELF file (D,P)
        PSARC/2008/198 elfwrap - wrap data in an ELF file
6678244 elfdump dynamic section sanity checking needs refinement
6679212 sgs use of SCCS id for versioning is obstacle to mercurial migration
```

```
6681761 lies, darn lies, and linker README files
6509323 Need to disable the Multiple Files loading - same name, different
        directories (or its stat() use)
6686889 ld.so.1 regression - bad pointer created with 6509323 integration
6695681 ldd(1) crashes when run from a chrooted environment
6516212 usr/src/cmd/sgs/libelf warlock targets should be fixed or abandoned
6678310 using LD_AUDIT, ld.so.1 calls shared library's .init before library is
        fully relocated (link-editor components only)
6699594 The ld command has a problem handling 'protected' mapfile keyword.
6699131 elfdump should display core file notes (D)
6702260 single threading .init/.fini sections breaks staroffice
6703919 boot hangs intermittently on x86 with onnv daily.0430 and on
6701798 ld can enter infinite loop processing bad mapfile
6706401 direct binding copy relocation fallback is insufficient for ild
        generated objects
6705846 multithreaded C++ application seems to get deadlocked in the dynamic
        linker code
6686343 ldd(1) - unused search path diagnosis should be enabled
6712292 ld.so.1 should fall back to an interposer for failed direct bindings
6716350 usr/src/cmd/sgs should be linted by nightly builds
6720509 usr/src/cmd/sgs/sgsdemangler should be removed
6617475 gas creates erroneous FILE symbols [was: ld.so.1 is reported as
        false positive by wsdiff]
6724311 dldump() mishandles R_AMD64_JUMP_SLOT relocations
6724774 elfdump -n doesn't print siginfo structure
6728555 Fix for amd64 aw (6617475) breaks pure gcc builds
6734598 ld(1) archive processing failure due to mismatched file descriptors (D)
6735939 ld(1) discarded symbol relocations errors (Studio and GNU).
6354160 Solaris linker includes more than one copy of code in binary when
        linking gnu object code
6744003 ld(1) could provide better argument processing diagnostics (D)
        PSARC 2008/583 add gld options to ld(1)
6749055 ld should generate GNU style VERSYM indexes for VERNEED records (D)
        PSARC/2008/603 ELF objects to adopt GNU-style Versym indexes
6752728 link-editor can enter UNDEF symbols in symbol sort sections
6756472 AOUT search path pruning (D)
--------------------------------------------------------------------------------

---------------------------------------------
Solaris Nevada (OpenSolaris 2009.06, snv_111)
---------------------------------------------
Bugid   Risk Synopsis
================================================================================

6754965 introduce the SF1_SUNW_ADDR32 bit in software capabilities (D)
        (link-editor components only)
        PSARC/2008/622 32-bit Address Restriction Software Capabilities Flag
6756953 customer requests that DT_CONFIG strings be honored for secure apps (D)
6765299 ld --version-script option not compatible with GNU ld (D)
6748160 problem with -zrescan (D)
        PSARC/2008/651 New ld archive rescan options
6763342 sloppy relocations need to get sloppier
6736890 PT_SUNWBSS should be disabled (D)
        PSARC/2008/715 PT_SUNWBSS removal
6772661 ldd/lddstub/ld.so.1 dump core in current nightly while processing
        libsoftcrypto_hwcap.so.1
6765931 mcs generates unlink(NULL) system calls
6775062 remove /usr/lib/libldstab.so (D)
6782977 ld segfaults after support lib version error sends bad args to vprintf()
6773695 ld -z nopartial can break non-pic objects
6778453 RTLD_GROUP prevents use of application defined malloc
6789925 64-bit applications with SF1_SUNW_ADDR32 require non-default starting
        address
6792906 ld -z nopartial fix breaks TLS
6686372 ld.so.1 should use mmapobj(2)
6726108 dlopen() performance could be improved.
```

```
6792836 ld is slow when processing GNU linkonce sections
6797468 ld.so.1: orphaned handles aren't processed correctly
6798676 ld.so.1: enters infinite loop with realloc/defragmentation logic
6237063 request extension to dl* family to provide segment bounds
        information (D)
        PSARC/2009/054 dlinfo(3c) - segment mapping retrieval
6800388 shstrtab can be sized incorrectly when -z ignore is used
6805009 ld.so.1: link map control list tear down leaves dangling pointer -
        pfinstall does it again.
6807050 GNU linkonce sections can create duplicate and incompatible
        eh_frame FDE entries
--------------------------------------------------------------------------------

--------------
Solaris Nevada
--------------
Bugid   Risk Synopsis
================================================================================
6813909 generalize eh_frame support to non-amd64 platforms
6801536 ld: mapfile processing oddities unveiled through mmapobj(2) observations
6802452 libelf shouldn't use MS_SYNC
6818012 nm tries to modify readonly segment and dumps core
6821646 xVM dom0 doesn't boot on daily.0324 and beyond
6822828 librtld_db can return RD_ERR before RD_NOMAPS, which compromises dbx
        expectations.
6821619 Solaris linkers need systematic approach to ELF OSABI (D)
        PSARC/2009/196 ELF objects to set OSABI / elfdump -O option
6827468 6801536 breaks 'ld -s' if there are weak/strong symbol pairs
6715578 AOUT (BCP) symbol lookup can be compromised with lazy loading.
6752883 ld.so.1 error message should be buffered (not sent to stderr).
6577982 ld.so.1 calls getpid() before it should when any LD_* are set
6831285 linker LD_DEBUG support needs improvements (D)
6806791 filter builds could be optimized (link-editor components only)
6823371 calloc() uses suboptimal memset() causing 15% regression in SpecCPU2006
        gcc code (link-editor components only)
6831308 ld.so.1: symbol rescanning does a little too much work
6837777 ld ordered section code uses too much memory and works too hard
6841199 Undo 10 year old workaround and use 64-bit ld on 32-bit objects
6784790 ld should examine archives to determine output object class/machine (D)
        PSARC/2009/305 ld -32 option
6849998 remove undocumented mapfile $SPECVERS and $NEED options
6851224 elf_getshnum() and elf_getshstrndx() incompatible with 2002 ELF gABI
        agreement (D)
        PSARC/2009/363 replace elf_getphnum, elf_getshnum, and elf_getshstrndx
6853809 ld.so.1: rescan fallback optimization is invalid
6854158 ld.so.1: interposition can be skipped because of incorrect
        caller/destination validation
6862967 rd_loadobj_iter() failing for core files
6856173 streams core dumps when compiled in 64bit with a very large static
        array size
6834197 ld pukes when given an empty plate
6516644 per-symbol filtering shouldn't be allowed in executables
6878605 ld should accept '%' syntax when matching input SHT_PROGBITS sections
6850768 ld option to autogenerate wrappers/interposers similar to GNU ld
        --wrap (D)
        PSARC/2009/493 ld -z wrap option
6888489 Null environment variables are not overriding crle(1) replaceable
        environment variables.
6885456 Need to implement GNU-ld behavior in construction of .init/.fini
        sections
6900241 ld should track SHT_GROUP sections by symbol name, not section name
6901773 Special handling of STT_SECTION group signature symbol for GNU objects
6901895 Failing asserts in ld update_osym() trying to build gcc 4.5 develpment
        head
6909523 core dump when run "LD_DEBUG=help ls" in non-English locale
6903688 mdb(1) can't resolve certain symbols in solaris10-branded processes
```

```
        from the global zone
6923449 elfdump misinterprets _init/_fini symbols in dynamic section test
6914728 Add dl_iterate_phdr() function to ld.so.1 (D)
        PSARC/2010/015 dl_iterate_phdr
6916788 ld version 2 mapfile syntax (D)
        PSARC/2009/688 Human readable and extensible ld mapfile syntax
6929607 ld generates incorrect VERDEF entries for ET_REL output objects
6924224 linker should ignore SUNW_dof when calculating the elf checksum
6918143 symbol capabilities (D)
        PSARC/2010/022 Linker-editors: Symbol Capabilities
6910387 .tdata and .tbss separation invalidates TLS program header information
6934123 elfdump -d coredumps on PA-RISC elf
6931044 ld should not allow SHT_PROGBITS .eh_frame sections on amd64 (D)
6931056 pvs -r output can include empty versions in output
6938628 ld.so.1 should produce diagnostics for all dl*() entry points
6938111 nm ‘No symbol table data’ message goes to stdout
6941727 ld relocation cache memory use is excessive
6932220 ld -z allextract skips objects that lack global symbols
6943772 Testing for a symbols existence with RTLD_PROBE is compromised by
        RTLD_BIND_NOW
        PSARC/2010/XXX Deferred symbol references
6943432 dlsym(RTLD_PROBE) should only bind to symbol definitions
6668759 an external method for determining whether an ELF dependency is optional
6954032 Support library with ld_open and -z allextract in snv_139 do not mix
6949596 wrong section alignment generated in joint compilation with shared
        library
6961755 ld.so.1’s -e arguments should take precedence over environment
        variables. (D)
6748925 moe returns wrong hwcap library in some circumstances
6916796 OSnet mapfiles should use version 2 link-editor syntax
6964517 OSnet mapfiles should use version 2 link-editor syntax (2nd pass)
6948720 SHT_INIT_ARRAY etc. section names don’t follow ELF gABI (D)
6962343 sgsmsg should use mkstemp() for temporary file creation
6965723 libsoftcrypto symbol capabilities rely on compiler generated
        capabilities - gcc failure (link-editor components only)
6952219 ld support for archives larger than 2 GB (D, P)
        PSARC/2010/224 Support for archives larger than 2 GB
6956152 dlclose() from an auditor can be fatal. Preinit/activity events should
        be more flexible. (D)
6971440 moe can core dump while processing libc.
6972234 sgs demo’s could use some cleanup
6935867 .dynamic could be readonly in sharable objects
6975290 ld mishandles GOT relocation against local ABS symbol
6972860 ld should provide user guidance to improve objects (D)
        PSARC/2010/312 Link-editor guidance
--------------------------------------------------------------------------------

--------------
Illumos
--------------
Bugid   Risk Synopsis
================================================================================

308     ld may misalign sections only preceded by empty sections
3265    link-editor builds bogus .eh_frame_hdr on ia32
#endif /* ! codereview */
```

```
**********************************************************
    8076 Thu Oct 11 22:38:20 2012
new/usr/src/cmd/sgs/tools/common/leb128.c
3265 link-editor builds bogus .eh_frame_hdr on ia32
**********************************************************
______unchanged_portion_omitted_

/*
 * Extract a DWARF encoded datum
 *
 * entry:
 *      data - Base of data buffer containing encoded bytes
 *      dotp - Address of variable containing index within data
 *              at which the desired datum starts.
 *      ehe_flags - DWARF encoding
 *      eident - ELF header e_ident[] array for object being processed
 *      sh_base - Base address of ELF section containing desired datum
 *      sh_offset - Offset relative to sh_base of desired datum.
 */
uint64_t
dwarf_ehe_extract(unsigned char *data, uint64_t *dotp, uint_t ehe_flags,
    unsigned char *eident, boolean_t frame_hdr, uint64_t sh_base,
    uint64_t sh_offset, uint64_t dbase)
    unsigned char *eident, uint64_t sh_base, uint64_t sh_offset)
{
        uint64_t    dot = *dotp;
        uint_t      lsb;
        uint_t      wordsize;
        uint_t      fsize;
        uint64_t    result;

        if (eident[EI_DATA] == ELFDATA2LSB)
                lsb = 1;
        else
                lsb = 0;

        if (eident[EI_CLASS] == ELFCLASS64)
                wordsize = 8;
        else
                wordsize = 4;

        switch (ehe_flags & 0x0f) {
        case DW_EH_PE_omit:
                return (0);
        case DW_EH_PE_absptr:
                fsize = wordsize;
                break;
        case DW_EH_PE_udata8:
        case DW_EH_PE_sdata8:
                fsize = 8;
                break;
        case DW_EH_PE_udata4:
        case DW_EH_PE_sdata4:
                fsize = 4;
                break;
        case DW_EH_PE_udata2:
        case DW_EH_PE_sdata2:
                fsize = 2;
                break;
        case DW_EH_PE_uleb128:
                return (uleb_extract(data, dotp));
        case DW_EH_PE_sleb128:
                return ((uint64_t)sleb_extract(data, dotp));
        default:
                return (0);
        }
```

```
        if (lsb) {
                /*
                 * Extract unaligned LSB formated data
                 */
                uint_t  cnt;

                result = 0;
                for (cnt = 0; cnt < fsize;
                    cnt++, dot++) {
                        uint64_t val;
                        val = data[dot];
                        result |= val << (cnt * 8);
                }
        } else {
                /*
                 * Extract unaligned MSB formated data
                 */
                uint_t  cnt;
                result = 0;
                for (cnt = 0; cnt < fsize;
                    cnt++, dot++) {
                        uint64_t        val;
                        val = data[dot];
                        result |= val << ((fsize - cnt - 1) * 8);
                }
        }
        /*
         * perform sign extension
         */
        if ((ehe_flags & DW_EH_PE_signed) &&
            (fsize < sizeof (uint64_t))) {
                int64_t sresult;
                uint_t  bitshift;
                sresult = result;
                bitshift = (sizeof (uint64_t) - fsize) * 8;
                sresult = (sresult << bitshift) >> bitshift;
                result = sresult;
        }

        /*
         * If value is relative to a base address, adjust it
         */
        if (result) {
        switch (ehe_flags & 0xf0) {
        case DW_EH_PE_pcrel:
                result += sh_base + sh_offset;
                break;

        /*
         * datarel is relative to .eh_frame_hdr if within .eh_frame,
         * but GOT if not.
         */
#endif /* ! codereview */
        case DW_EH_PE_datarel:
                if (frame_hdr)
#endif /* ! codereview */
                        result += sh_base;
                else
                        result += dbase;
#endif /* ! codereview */
                break;
        }

        /* Truncate the result to its specified size */
        result = (result << ((sizeof (uint64_t) - fsize) * 8)) >>
```

```
                ((sizeof (uint64_t) - fsize) * 8);

        }
        *dotp = dot;
        return (result);
}
```
______***unchanged_portion_omitted_***

```
**********************************************************
   10250 Thu Oct 11 22:38:20 2012
new/usr/src/head/link.h
3263 link.h should work in a largefile environment
**********************************************************
/*
 * 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) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
 */

#ifndef _LINK_H
#define _LINK_H

#include <sys/link.h>

#ifndef _ASM
#include <elf.h>
#include <libelf.h>
#include <sys/types.h>
#include <dlfcn.h>
#endif

#ifdef  __cplusplus
extern "C" {
#endif

#ifndef _ASM
/*
 * ld support library calls.
 *
 * These cannot be used in a 32bit large file capable environment,
 * thanks to the use of libelf.
 * ld support library calls
 */
#if !defined(_ILP32) || _FILE_OFFSET_BITS != 64
#endif /* ! codereview */
#ifdef __STDC__
#include <libelf.h>
#endif /* ! codereview */
extern uint_t   ld_version(uint_t);
extern void     ld_input_done(uint_t *);

extern void     ld_start(const char *, const Elf32_Half, const char *);
extern void     ld_atexit(int);
extern void     ld_open(const char **, const char **, int *, int, Elf **,
                        Elf *, size_t, const Elf_Kind);
extern void     ld_file(const char *, const Elf_Kind, int, Elf *);
```

```
extern void     ld_input_section(const char *, Elf32_Shdr **, Elf32_Word,
                        Elf_Data *, Elf *, uint_t *);
extern void     ld_section(const char *, Elf32_Shdr *, Elf32_Word,
                        Elf_Data *, Elf *);

#if defined(_LP64) || defined(_LONGLONG_TYPE)
extern void     ld_start64(const char *, const Elf64_Half, const char *);
extern void     ld_atexit64(int);
extern void     ld_open64(const char **, const char **, int *, int, Elf **,
                        Elf *, size_t, const Elf_Kind);
extern void     ld_file64(const char *, const Elf_Kind, int, Elf *);
extern void     ld_input_section64(const char *, Elf64_Shdr **, Elf64_Word,
                        Elf_Data *, Elf *, uint_t *);
extern void     ld_section64(const char *, Elf64_Shdr *, Elf64_Word,
                        Elf_Data *, Elf *);

#endif /* (defined(_LP64) || defined(_LONGLONG_TYPE) */
#else
extern void     ld_version();
extern void     ld_input_done();

extern void     ld_start();
extern void     ld_atexit();
extern void     ld_open();
extern void     ld_file();
extern void     ld_input_section();
extern void     ld_section();

#if defined(_LP64) || defined(_LONGLONG_TYPE)
extern void     ld_start64();
extern void     ld_atexit64();
extern void     ld_open64();
extern void     ld_file64();
extern void     ld_input_section64();
extern void     ld_section64();

#endif /* (defined(_LP64) || defined(_LONGLONG_TYPE) */
#endif /* __STDC__ */
#endif /* !defined(_ILP32) || _FILE_OFFSET_BITS != 64 */
#endif /* ! codereview */

/*
 * ld_version() version values.
 */
#define LD_SUP_VNONE    0
#define LD_SUP_VERSION1 1
#define LD_SUP_VERSION2 2
#define LD_SUP_VERSION3 3
#define LD_SUP_VCURRENT LD_SUP_VERSION3

/*
 * Flags passed to ld support calls.
 */
#define LD_SUP_DERIVED          0x1     /* derived filename */
#define LD_SUP_INHERITED        0x2     /* file inherited from .so DT_NEEDED */
#define LD_SUP_EXTRACTED        0x4     /* file extracted from archive */
#endif

/*
 * Runtime link-map identifiers.
 */
#define LM_ID_BASE              0x00
#define LM_ID_LDSO              0x01
#define LM_ID_NUM               2

#define LM_ID_BRAND             0xfd    /* brand emulation linkmap objs */
```

```
#define LM_ID_NONE              0xfe    /* no link map specified */
#define LM_ID_NEWLM             0xff    /* create a new link-map */

/*
 * Runtime Link-Edit Auditing.
 */
#define LAV_NONE                0
#define LAV_VERSION1            1
#define LAV_VERSION2            2
#define LAV_VERSION3            3
#define LAV_VERSION4            4
#define LAV_VERSION5            5
#define LAV_CURRENT             LAV_VERSION5
#define LAV_NUM                 6

/*
 * Flags that can be or'd into the la_objopen() return code
 */
#define LA_FLG_BINDTO           0x0001  /* audit symbinds TO this object */
#define LA_FLG_BINDFROM         0x0002  /* audit symbinding FROM this object */

/*
 * Flags that can be or'd into the 'flags' argument of la_symbind()
 */
#define LA_SYMB_NOPLTENTER      0x0001  /* disable pltenter for this symbol */
#define LA_SYMB_NOPLTEXIT       0x0002  /* disable pltexit for this symbol */
#define LA_SYMB_STRUCTCALL      0x0004  /* this function call passes a */
                                        /*      structure as it's return code */
#define LA_SYMB_DLSYM           0x0008  /* this symbol bindings is due to */
                                        /*      a call to dlsym() */
#define LA_SYMB_ALTVALUE        0x0010  /* alternate symbol binding returned */
                                        /*      by la_symbind() */

/*
 * Flags that describe the object passed to la_objsearch()
 */
#define LA_SER_ORIG             0x001   /* original (needed) name */
#define LA_SER_LIBPATH          0x002   /* LD_LIBRARY_PATH entry prepended */
#define LA_SER_RUNPATH          0x004   /* runpath entry prepended */
#define LA_SER_CONFIG           0x008   /* configuration entry prepended */
#define LA_SER_DEFAULT          0x040   /* default path prepended */
#define LA_SER_SECURE           0x080   /* default (secure) path prepended */

#define LA_SER_MASK             0xfff   /* mask of known flags */

/*
 * Flags that describe the la_activity()
 */
#define LA_ACT_CONSISTENT       0x00    /* add/deletion of objects complete */
#define LA_ACT_ADD              0x01    /* objects being added */
#define LA_ACT_DELETE           0x02    /* objects being deleted */
#define LA_ACT_MAX              3


#ifndef _KERNEL
#ifndef _ASM

#if defined(_LP64)
typedef long    lagreg_t;
#else
typedef int     lagreg_t;
#endif

struct _la_sparc_regs {
        lagreg_t        lr_rego0;
        lagreg_t        lr_rego1;
```

```
        lagreg_t        lr_rego2;
        lagreg_t        lr_rego3;
        lagreg_t        lr_rego4;
        lagreg_t        lr_rego5;
        lagreg_t        lr_rego6;
        lagreg_t        lr_rego7;
};

#if defined(_LP64)
typedef struct _la_sparc_regs   La_sparcv9_regs;
typedef struct {
        lagreg_t        lr_rsp;
        lagreg_t        lr_rbp;
        lagreg_t        lr_rdi;     /* arg1 */
        lagreg_t        lr_rsi;     /* arg2 */
        lagreg_t        lr_rdx;     /* arg3 */
        lagreg_t        lr_rcx;     /* arg4 */
        lagreg_t        lr_r8;      /* arg5 */
        lagreg_t        lr_r9;      /* arg6 */
} La_amd64_regs;
#else
typedef struct _la_sparc_regs   La_sparcv8_regs;
typedef struct {
        lagreg_t        lr_esp;
        lagreg_t        lr_ebp;
} La_i86_regs;
#endif

#if     !defined(_SYS_INT_TYPES_H)
#if     defined(_LP64) || defined(_I32LPx)
typedef unsigned long           uintptr_t;
#else
typedef unsigned int            uintptr_t;
#endif
#endif


#ifdef  __STDC__
extern uint_t           la_version(uint_t);
extern void             la_activity(uintptr_t *, uint_t);
extern void             la_preinit(uintptr_t *);
extern char             *la_objsearch(const char *, uintptr_t *, uint_t);
extern uint_t           la_objopen(Link_map *, Lmid_t, uintptr_t *);
extern uint_t           la_objclose(uintptr_t *);
extern int              la_objfilter(uintptr_t *, const char *, uintptr_t *,
                                uint_t);
#if     defined(_LP64)
extern uintptr_t        la_amd64_pltenter(Elf64_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, La_amd64_regs *,   uint_t *,
                                const char *);
extern uintptr_t        la_symbind64(Elf64_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, uint_t *, const char *);
extern uintptr_t        la_sparcv9_pltenter(Elf64_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, La_sparcv9_regs *, uint_t *,
                                const char *);
extern uintptr_t        la_pltexit64(Elf64_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, uintptr_t, const char *);
#else  /* !defined(_LP64) */
extern uintptr_t        la_symbind32(Elf32_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, uint_t *);
extern uintptr_t        la_sparcv8_pltenter(Elf32_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, La_sparcv8_regs *, uint_t *);
extern uintptr_t        la_i86_pltenter(Elf32_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, La_i86_regs *, uint_t *);
extern uintptr_t        la_pltexit(Elf32_Sym *, uint_t, uintptr_t *,
                                uintptr_t *, uintptr_t);
```

```
#endif /* _LP64 */
#else  /* __STDC__ */
extern uint_t           la_version();
extern void             la_preinit();
extern uint_t           la_objopen();
extern uint_t           la_objclose();
extern int              la_objfilter();
#if     defined(_LP64)
extern uintptr_t        la_sparcv9_pltenter();
extern uintptr_t        la_pltexit64();
extern uintptr_t        la_symbind64();
#else  /* _ILP32 */
extern uintptr_t        la_sparcv8_pltenter();
extern uintptr_t        la_i86_pltenter();
extern uintptr_t        la_pltexit();
extern uintptr_t        la_symbind32();
#endif /* _LP64 */
#endif /* __STDC__ */

#ifdef  __STDC__
/*
 * The ElfW() macro is a GNU/Linux feature, provided as support for
 * the dl_phdr_info structure used by dl_phdr_iterate(), which also
 * originated under Linux. Given an ELF data type, without the ElfXX_
 * prefix, it supplies the appropriate prefix (Elf32_ or Elf64_) for
 * the ELFCLASS of the code being compiled.
 *
 * Note that ElfW() is not suitable in situations in which the ELFCLASS
 * of the code being compiled does not match that of the objects that
 * code is intended to operate on (e.g. a 32-bit link-editor building
 * a 64-bit object). The macros defined in <sys/machelf.h> are
 * recommended in such cases.
 */
#ifdef _LP64
#define ElfW(type)      Elf64_ ## type
#else
#define ElfW(type)      Elf32_ ## type
#endif

/*
 * The callback function to dl_interate_phdr() receives a pointer
 * to a structure of this type.
 *
 * dlpi_addr is defined such that the address of any segment in
 * the program header array can be calculated as:
 *
 *      addr == info->dlpi_addr + info->dlpi_phdr[x].p_vaddr;
 *
 * It is therefore 0 for ET_EXEC objects, and the base address at
 * which the object is mapped otherwise.
 */
struct dl_phdr_info {
        ElfW(Addr)              dlpi_addr;      /* Base address of object */
        const char              *dlpi_name;     /* Null-terminated obj name */
        const ElfW(Phdr)        *dlpi_phdr;     /* Ptr to ELF program hdr arr */
        ElfW(Half)              dlpi_phnum;     /* # of items in dlpi_phdr[] */

        /*
         * Note: Following members were introduced after the first version
         * of this structure was available.  The dl_iterate_phdr() callback
         * function is passed a 'size' argument giving the size of the info
         * structure, and must compare that size to the offset of these fields
         * before accessing them to ensure that they are present.
         */

        /* Incremented when a new object is mapped into the process */
```

```
        u_longlong_t            dlpi_adds;
        /* Incremented when an object is unmapped from the process */
        u_longlong_t            dlpi_subs;
};

extern  int dl_iterate_phdr(int (*)(struct dl_phdr_info *, size_t, void *),
            void *);
#endif /* __STDC__ */

#endif  /* _ASM */
#endif /* _KERNEL */


#ifdef __cplusplus
}
#endif

#endif  /* _LINK_H */
```