```
**********************************************************
   15201 Fri Jan 30 17:30:43 2015
new/usr/src/cmd/file/elf_read.c
5578 file(1) should validate Elf_Shdr->sh_name
**********************************************************
______unchanged_portion_omitted_

/*
 * process_shdr:        Read Section Headers to attempt to get HW/SW
 *                      capabilities by looking at the SUNW_cap
 *                      section and set string in Elf_Info.
 *                      Also look for symbol tables and debug
 *                      information sections. Set the "stripped" field
 *                      in Elf_Info with corresponding flags.
 */
static int
process_shdr(Elf_Info *EI)
{
        int             capn, mac;
        int             i, j, idx;
        FILE_ELF_OFF_T  cap_off;
        FILE_ELF_SIZE_T csize;
        char            *strtab;
        size_t          strtab_sz;
        char            *section_name;
        Elf_Cap         Chdr;
        Elf_Shdr        *shdr = &EI_Shdr;


        csize = sizeof (Elf_Cap);
        mac = EI_Ehdr.e_machine;

        /* if there are no sections, return success anyway */
        if (EI_Ehdr.e_shoff == 0 && EI_Ehdr_shnum == 0)
                return (ELF_READ_OKAY);

        /* read section names from String Section */
        if (get_shdr(EI, EI_Ehdr_shstrndx) == ELF_READ_FAIL)
                return (ELF_READ_FAIL);

        if ((strtab = malloc(shdr->sh_size)) == NULL)
        if ((section_name = malloc(shdr->sh_size)) == NULL)
                return (ELF_READ_FAIL);

        if (pread64(EI->elffd, strtab, shdr->sh_size, shdr->sh_offset)
        if (pread64(EI->elffd, section_name, shdr->sh_size, shdr->sh_offset)
            != shdr->sh_size)
                return (ELF_READ_FAIL);

        strtab_sz = shdr->sh_size;

#endif /* ! codereview */
        /* read all the sections and process them */
        for (idx = 1, i = 0; i < EI_Ehdr_shnum; idx++, i++) {
                char *shnam;
                char *str;

                if (get_shdr(EI, i) == ELF_READ_FAIL)
                        return (ELF_READ_FAIL);

                if (shdr->sh_type == SHT_NULL) {
                        idx--;
                        continue;
                }

                cap_off = shdr->sh_offset;
```

```
                if (shdr->sh_type == SHT_SUNW_cap) {
                        char capstr[128];

                        if (shdr->sh_size == 0 || shdr->sh_entsize == 0) {
                                (void) fprintf(stderr, ELF_ERR_ELFCAP1,
                                    File, EI->file);
                                return (ELF_READ_FAIL);
                        }
                        capn = (shdr->sh_size / shdr->sh_entsize);
                        for (j = 0; j < capn; j++) {
                                /*
                                 * read cap and xlate the values
                                 */
                                if (pread64(EI->elffd, &Chdr, csize, cap_off)
                                    != csize ||
                                    file_xlatetom(ELF_T_CAP, (char *)&Chdr)
                                    == 0) {
                                        (void) fprintf(stderr, ELF_ERR_ELFCAP2,
                                            File, EI->file);
                                        return (ELF_READ_FAIL);
                                }

                                cap_off += csize;

                                /*
                                 * Each capability group is terminated with
                                 * CA_SUNW_NULL.  Groups other than the first
                                 * represent symbol capabilities, and aren't
                                 * interesting here.
                                 */
                                if (Chdr.c_tag == CA_SUNW_NULL)
                                        break;

                                (void) elfcap_tag_to_str(ELFCAP_STYLE_UC,
                                    Chdr.c_tag, Chdr.c_un.c_val, capstr,
                                    sizeof (capstr), ELFCAP_FMT_SNGSPACE,
                                    mac);

                                if ((*EI->cap_str != '\0') && (*capstr != '\0'))
                                        (void) strlcat(EI->cap_str, " ",
                                            sizeof (EI->cap_str));

                                (void) strlcat(EI->cap_str, capstr,
                                    sizeof (EI->cap_str));
                        }
                }

                /*
                 * Definition time:
                 *      - "not stripped" means that an executable file
                 *      contains a Symbol Table (.symtab)
                 *      - "stripped" means that an executable file
                 *      does not contain a Symbol Table.
                 * When strip -l or strip -x is run, it strips the
                 * debugging information (.line section name (strip -l),
                 * .line, .debug*, .stabs*, .dwarf* section names
                 * and SHT_SUNW_DEBUGSTR and SHT_SUNW_DEBUG
                 * section types (strip -x), however the Symbol
                 * Table will still be present.
                 * Therefore, if
                 *      - No Symbol Table present, then report
                 *              "stripped"
                 *      - Symbol Table present with debugging
                 *      information (line number or debug section names,
                 *      or SHT_SUNW_DEBUGSTR or SHT_SUNW_DEBUG section
                 *      types) then report:
```

```
                 *              "not stripped"
                 *      - Symbol Table present with no debugging
                 *      information (line number or debug section names,
                 *      or SHT_SUNW_DEBUGSTR or SHT_SUNW_DEBUG section
                 *      types) then report:
                 *              "not stripped, no debugging information
                 *              available"
                 */
                if ((EI->stripped & E_NOSTRIP) == E_NOSTRIP)
                        continue;

                if (!(EI->stripped & E_SYMTAB) &&
                    (shdr->sh_type == SHT_SYMTAB)) {
                        EI->stripped |= E_SYMTAB;
                        continue;
                }

                if (shdr->sh_name >= strtab_sz)
                        shnam = NULL;
                else
                        shnam = &strtab[shdr->sh_name];
                str = &section_name[shdr->sh_name];

                if (!(EI->stripped & E_DBGINF) &&
                    ((shdr->sh_type == SHT_SUNW_DEBUG) ||
                    (shdr->sh_type == SHT_SUNW_DEBUGSTR) ||
                    (shnam != NULL && is_in_list(shnam)))) {
                    (is_in_list(str)))) {
                        EI->stripped |= E_DBGINF;
                }
        }
        free(strtab);
        free(section_name);

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