```
**********************************************************
   65298 Fri Nov 18 16:39:48 2016
new/usr/src/cmd/sgs/libld/common/args.c
7594 ld -zaslr should accept Solaris-compatible values
**********************************************************
______unchanged_portion_omitted_

 965 static int      optitle = 0;
 966 /*
 967  * Parsing options pass1 for process_flags().
 968  */
 969 static uintptr_t
 970 parseopt_pass1(Ofl_desc *ofl, int argc, char **argv, int *usage)
 971 {
 972         int     c, ndx = optind;

 974         /*
 975          * The -32, -64 and -ztarget options are special, in that we validate
 976          * them, but otherwise ignore them. libld.so (this code) is called
 977          * from the ld front end program. ld has already examined the
 978          * arguments to determine the output class and machine type of the
 979          * output object, as reflected in the version (32/64) of ld_main()
 980          * that was called and the value of the 'mach' argument passed.
 981          * By time execution reaches this point, these options have already
 982          * been seen and acted on.
 983          */
 984         while ((c = ld_getopt(ofl->ofl_lml, ndx, argc, argv)) != -1) {

 986                 switch (c) {
 987                 case '3':
 988                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

 990                         /*
 991                          * -32 is processed by ld to determine the output class.
 992                          * Here we sanity check the option incase some other
 993                          * -3* option is mistakenly passed to us.
 994                          */
 995                         if (optarg[0] != '2')
 996                                 ld_eprintf(ofl, ERR_FATAL,
 997                                     MSG_INTL(MSG_ARG_ILLEGAL),
 998                                     MSG_ORIG(MSG_ARG_3), optarg);
 999                         continue;

1001                 case '6':
1002                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

1004                         /*
1005                          * -64 is processed by ld to determine the output class.
1006                          * Here we sanity check the option incase some other
1007                          * -6* option is mistakenly passed to us.
1008                          */
1009                         if (optarg[0] != '4')
1010                                 ld_eprintf(ofl, ERR_FATAL,
1011                                     MSG_INTL(MSG_ARG_ILLEGAL),
1012                                     MSG_ORIG(MSG_ARG_6), optarg);
1013                         continue;

1015                 case 'a':
1016                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1017                         aflag = TRUE;
1018                         break;

1020                 case 'b':
1021                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1022                         bflag = TRUE;
```



```
1024                         /*
1025                          * This is a hack, and may be undone later.
1026                          * The -b option is only used to build the Unix
1027                          * kernel and its related kernel-mode modules.
1028                          * We do not want those files to get a .SUNW_ldynsym
1029                          * section. At least for now, the kernel makes no
1030                          * use of .SUNW_ldynsym, and we do not want to use
1031                          * the space to hold it. Therefore, we overload
1032                          * the use of -b to also imply -znoldynsym.
1033                          */
1034                         ofl->ofl_flags |= FLG_OF_NOLDYNSYM;
1035                         break;

1037                 case 'c':
1038                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1039                         if (ofl->ofl_config)
1040                                 ld_eprintf(ofl, ERR_WARNING_NF,
1041                                     MSG_INTL(MSG_ARG_MTONCE),
1042                                     MSG_ORIG(MSG_ARG_C));
1043                         else
1044                                 ofl->ofl_config = optarg;
1045                         break;

1047                 case 'C':
1048                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1049                         demangle_flag = 1;
1050                         break;

1052                 case 'd':
1053                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1054                         if ((optarg[0] == 'n') && (optarg[1] == '\0')) {
1055                                 if (dflag != SET_UNKNOWN)
1056                                         ld_eprintf(ofl, ERR_WARNING_NF,
1057                                             MSG_INTL(MSG_ARG_MTONCE),
1058                                             MSG_ORIG(MSG_ARG_D));
1059                                 else
1060                                         dflag = SET_FALSE;
1061                         } else if ((optarg[0] == 'y') && (optarg[1] == '\0')) {
1062                                 if (dflag != SET_UNKNOWN)
1063                                         ld_eprintf(ofl, ERR_WARNING_NF,
1064                                             MSG_INTL(MSG_ARG_MTONCE),
1065                                             MSG_ORIG(MSG_ARG_D));
1066                                 else
1067                                         dflag = SET_TRUE;
1068                         } else {
1069                                 ld_eprintf(ofl, ERR_FATAL,
1070                                     MSG_INTL(MSG_ARG_ILLEGAL),
1071                                     MSG_ORIG(MSG_ARG_D), optarg);
1072                         }
1073                         break;

1075                 case 'e':
1076                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1077                         if (ofl->ofl_entry)
1078                                 ld_eprintf(ofl, ERR_WARNING_NF,
1079                                     MSG_INTL(MSG_MARG_MTONCE),
1080                                     MSG_INTL(MSG_MARG_ENTRY));
1081                         else
1082                                 ofl->ofl_entry = (void *)optarg;
1083                         break;

1085                 case 'f':
1086                         DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1087                         if (ofl->ofl_filtees &&
1088                             (!(ofl->ofl_flags & FLG_OF_AUX))) {
1089                                 ld_eprintf(ofl, ERR_FATAL,
```

```
1090                                    MSG_INTL(MSG_MARG_INCOMP),
1091                                    MSG_INTL(MSG_MARG_FILTER_AUX),
1092                                    MSG_INTL(MSG_MARG_FILTER));
1093                            } else {
1094                                    if ((ofl->ofl_filtees =
1095                                        add_string(ofl->ofl_filtees, optarg)) ==
1096                                        (const char *)S_ERROR)
1097                                            return (S_ERROR);
1098                                    ofl->ofl_flags |= FLG_OF_AUX;
1099                            }
1100                            break;

1102                    case 'F':
1103                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1104                            if (ofl->ofl_filtees &&
1105                                (ofl->ofl_flags & FLG_OF_AUX)) {
1106                                    ld_eprintf(ofl, ERR_FATAL,
1107                                        MSG_INTL(MSG_MARG_INCOMP),
1108                                        MSG_INTL(MSG_MARG_FILTER),
1109                                        MSG_INTL(MSG_MARG_FILTER_AUX));
1110                            } else {
1111                                    if ((ofl->ofl_filtees =
1112                                        add_string(ofl->ofl_filtees, optarg)) ==
1113                                        (const char *)S_ERROR)
1114                                            return (S_ERROR);
1115                            }
1116                            break;

1118                    case 'h':
1119                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1120                            if (ofl->ofl_soname)
1121                                    ld_eprintf(ofl, ERR_WARNING_NF,
1122                                        MSG_INTL(MSG_MARG_MTONCE),
1123                                        MSG_INTL(MSG_MARG_SONAME));
1124                            else
1125                                    ofl->ofl_soname = (const char *)optarg;
1126                            break;

1128                    case 'i':
1129                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1130                            ofl->ofl_flags |= FLG_OF_IGNENV;
1131                            break;

1133                    case 'I':
1134                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1135                            if (ofl->ofl_interp)
1136                                    ld_eprintf(ofl, ERR_WARNING_NF,
1137                                        MSG_INTL(MSG_ARG_MTONCE),
1138                                        MSG_ORIG(MSG_ARG_CI));
1139                            else
1140                                    ofl->ofl_interp = (const char *)optarg;
1141                            break;

1143                    case 'l':
1144                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1145                            /*
1146                             * For now, count any library as a shared object.  This
1147                             * is used to size the internal symbol cache.  This
1148                             * value is recalculated later on actual file processing
1149                             * to get an accurate shared object count.
1150                             */
1151                            ofl->ofl_soscnt++;
1152                            break;

1154                    case 'm':
1155                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
```



```
1156                            ofl->ofl_flags |= FLG_OF_GENMAP;
1157                            break;

1159                    case 'o':
1160                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1161                            if (ofl->ofl_name)
1162                                    ld_eprintf(ofl, ERR_WARNING_NF,
1163                                        MSG_INTL(MSG_MARG_MTONCE),
1164                                        MSG_INTL(MSG_MARG_OUTFILE));
1165                            else
1166                                    ofl->ofl_name = (const char *)optarg;
1167                            break;

1169                    case 'p':
1170                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

1172                            /*
1173                             * Multiple instances of this option may occur.  Each
1174                             * additional instance is effectively concatenated to
1175                             * the previous separated by a colon.
1176                             */
1177                            if (*optarg != '\0') {
1178                                    if ((ofl->ofl_audit =
1179                                        add_string(ofl->ofl_audit,
1180                                        optarg)) == (const char *)S_ERROR)
1181                                            return (S_ERROR);
1182                            }
1183                            break;

1185                    case 'P':
1186                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

1188                            /*
1189                             * Multiple instances of this option may occur.  Each
1190                             * additional instance is effectively concatenated to
1191                             * the previous separated by a colon.
1192                             */
1193                            if (*optarg != '\0') {
1194                                    if ((ofl->ofl_depaudit =
1195                                        add_string(ofl->ofl_depaudit,
1196                                        optarg)) == (const char *)S_ERROR)
1197                                            return (S_ERROR);
1198                            }
1199                            break;

1201                    case 'r':
1202                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1203                            rflag = TRUE;
1204                            break;

1206                    case 'R':
1207                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

1209                            /*
1210                             * Multiple instances of this option may occur.  Each
1211                             * additional instance is effectively concatenated to
1212                             * the previous separated by a colon.
1213                             */
1214                            if (*optarg != '\0') {
1215                                    if ((ofl->ofl_rpath =
1216                                        add_string(ofl->ofl_rpath,
1217                                        optarg)) == (const char *)S_ERROR)
1218                                            return (S_ERROR);
1219                            }
1220                            break;
```

```
1222                case 's':
1223                        DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1224                        sflag = TRUE;
1225                        break;

1227                case 't':
1228                        DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1229                        ofl->ofl_flags |= FLG_OF_NOWARN;
1230                        break;

1232                case 'u':
1233                        DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1234                        break;

1236                case 'z':
1237                        DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));

1239                        /*
1240                         * For specific help, print our usage message and exit
1241                         * immediately to ensure a 0 return code.
1242                         */
1243                        if (strncmp(optarg, MSG_ORIG(MSG_ARG_HELP),
1244                            MSG_ARG_HELP_SIZE) == 0) {
1245                                usage_mesg(TRUE);
1246                                exit(0);
1247                        }

1249                        /*
1250                         * For some options set a flag - further consistancy
1251                         * checks will be carried out in check_flags().
1252                         */
1253                        if ((strncmp(optarg, MSG_ORIG(MSG_ARG_LD32),
1254                            MSG_ARG_LD32_SIZE) == 0) ||
1255                            (strncmp(optarg, MSG_ORIG(MSG_ARG_LD64),
1256                            MSG_ARG_LD64_SIZE) == 0)) {
1257                                if (createargv(ofl, usage) == S_ERROR)
1258                                        return (S_ERROR);

1260                        } else if (
1261                            strcmp(optarg, MSG_ORIG(MSG_ARG_DEFS)) == 0) {
1262                                if (zdflag != SET_UNKNOWN)
1263                                        ld_eprintf(ofl, ERR_WARNING_NF,
1264                                            MSG_INTL(MSG_ARG_MTONCE),
1265                                            MSG_ORIG(MSG_ARG_ZDEFNODEF));
1266                                else
1267                                        zdflag = SET_TRUE;
1268                                ofl->ofl_guideflags |= FLG_OFG_NO_DEFS;
1269                        } else if (strcmp(optarg,
1270                            MSG_ORIG(MSG_ARG_NODEFS)) == 0) {
1271                                if (zdflag != SET_UNKNOWN)
1272                                        ld_eprintf(ofl, ERR_WARNING_NF,
1273                                            MSG_INTL(MSG_ARG_MTONCE),
1274                                            MSG_ORIG(MSG_ARG_ZDEFNODEF));
1275                                else
1276                                        zdflag = SET_FALSE;
1277                                ofl->ofl_guideflags |= FLG_OFG_NO_DEFS;
1278                        } else if (strcmp(optarg,
1279                            MSG_ORIG(MSG_ARG_TEXT)) == 0) {
1280                                if (ztflag &&
1281                                    (ztflag != MSG_ORIG(MSG_ARG_ZTEXT)))
1282                                        ld_eprintf(ofl, ERR_FATAL,
1283                                            MSG_INTL(MSG_ARG_INCOMP),
1284                                            MSG_ORIG(MSG_ARG_ZTEXT),
1285                                            ztflag);
1286                                ztflag = MSG_ORIG(MSG_ARG_ZTEXT);
1287                        } else if (strcmp(optarg,
```



```
1288                            MSG_ORIG(MSG_ARG_TEXTOFF)) == 0) {
1289                                if (ztflag &&
1290                                    (ztflag != MSG_ORIG(MSG_ARG_ZTEXTOFF)))
1291                                        ld_eprintf(ofl, ERR_FATAL,
1292                                            MSG_INTL(MSG_ARG_INCOMP),
1293                                            MSG_ORIG(MSG_ARG_ZTEXTOFF),
1294                                            ztflag);
1295                                ztflag = MSG_ORIG(MSG_ARG_ZTEXTOFF);
1296                        } else if (strcmp(optarg,
1297                            MSG_ORIG(MSG_ARG_TEXTWARN)) == 0) {
1298                                if (ztflag &&
1299                                    (ztflag != MSG_ORIG(MSG_ARG_ZTEXTWARN)))
1300                                        ld_eprintf(ofl, ERR_FATAL,
1301                                            MSG_INTL(MSG_ARG_INCOMP),
1302                                            MSG_ORIG(MSG_ARG_ZTEXTWARN),
1303                                            ztflag);
1304                                ztflag = MSG_ORIG(MSG_ARG_ZTEXTWARN);

1306                        /*
1307                         * For other options simply set the ofl flags directly.
1308                         */
1309                        } else if (strcmp(optarg,
1310                            MSG_ORIG(MSG_ARG_RESCAN)) == 0) {
1311                                ofl->ofl_flags1 |= FLG_OF1_RESCAN;
1312                        } else if (strcmp(optarg,
1313                            MSG_ORIG(MSG_ARG_ABSEXEC)) == 0) {
1314                                ofl->ofl_flags1 |= FLG_OF1_ABSEXEC;
1315                        } else if (strcmp(optarg,
1316                            MSG_ORIG(MSG_ARG_LOADFLTR)) == 0) {
1317                                zlflag = TRUE;
1318                        } else if (strcmp(optarg,
1319                            MSG_ORIG(MSG_ARG_NORELOC)) == 0) {
1320                                ofl->ofl_dtflags_1 |= DF_1_NORELOC;
1321                        } else if (strcmp(optarg,
1322                            MSG_ORIG(MSG_ARG_NOVERSION)) == 0) {
1323                                ofl->ofl_flags |= FLG_OF_NOVERSEC;
1324                        } else if (strcmp(optarg,
1325                            MSG_ORIG(MSG_ARG_MULDEFS)) == 0) {
1326                                ofl->ofl_flags |= FLG_OF_MULDEFS;
1327                        } else if (strcmp(optarg,
1328                            MSG_ORIG(MSG_ARG_REDLOCSYM)) == 0) {
1329                                ofl->ofl_flags |= FLG_OF_REDLSYM;
1330                        } else if (strcmp(optarg,
1331                            MSG_ORIG(MSG_ARG_INITFIRST)) == 0) {
1332                                ofl->ofl_dtflags_1 |= DF_1_INITFIRST;
1333                        } else if (strcmp(optarg,
1334                            MSG_ORIG(MSG_ARG_NODELETE)) == 0) {
1335                                ofl->ofl_dtflags_1 |= DF_1_NODELETE;
1336                        } else if (strcmp(optarg,
1337                            MSG_ORIG(MSG_ARG_NOPARTIAL)) == 0) {
1338                                ofl->ofl_flags1 |= FLG_OF1_NOPARTI;
1339                        } else if (strcmp(optarg,
1340                            MSG_ORIG(MSG_ARG_NOOPEN)) == 0) {
1341                                ofl->ofl_dtflags_1 |= DF_1_NOOPEN;
1342                        } else if (strcmp(optarg,
1343                            MSG_ORIG(MSG_ARG_NOW)) == 0) {
1344                                ofl->ofl_dtflags_1 |= DF_1_NOW;
1345                                ofl->ofl_dtflags |= DF_BIND_NOW;
1346                        } else if (strcmp(optarg,
1347                            MSG_ORIG(MSG_ARG_ORIGIN)) == 0) {
1348                                ofl->ofl_dtflags_1 |= DF_1_ORIGIN;
1349                                ofl->ofl_dtflags |= DF_ORIGIN;
1350                        } else if (strcmp(optarg,
1351                            MSG_ORIG(MSG_ARG_NODEFAULTLIB)) == 0) {
1352                                ofl->ofl_dtflags_1 |= DF_1_NODEFLIB;
1353                        } else if (strcmp(optarg,
```

```
1354                            MSG_ORIG(MSG_ARG_NODUMP)) == 0) {
1355                                ofl->ofl_dtflags_1 |= DF_1_NODUMP;
1356                        } else if (strcmp(optarg,
1357                            MSG_ORIG(MSG_ARG_ENDFILTEE)) == 0) {
1358                                ofl->ofl_dtflags_1 |= DF_1_ENDFILTEE;
1359                        } else if (strcmp(optarg,
1360                            MSG_ORIG(MSG_ARG_VERBOSE)) == 0) {
1361                                ofl->ofl_flags |= FLG_OF_VERBOSE;
1362                        } else if (strcmp(optarg,
1363                            MSG_ORIG(MSG_ARG_COMBRELOC)) == 0) {
1364                                ofl->ofl_flags |= FLG_OF_COMREL;
1365                        } else if (strcmp(optarg,
1366                            MSG_ORIG(MSG_ARG_NOCOMBRELOC)) == 0) {
1367                                ofl->ofl_flags |= FLG_OF_NOCOMREL;
1368                        } else if (strcmp(optarg,
1369                            MSG_ORIG(MSG_ARG_NOCOMPSTRTAB)) == 0) {
1370                                ofl->ofl_flags1 |= FLG_OF1_NCSTTAB;
1371                        } else if (strcmp(optarg,
1372                            MSG_ORIG(MSG_ARG_NOINTERP)) == 0) {
1373                                ofl->ofl_flags1 |= FLG_OF1_NOINTRP;
1374                        } else if (strcmp(optarg,
1375                            MSG_ORIG(MSG_ARG_INTERPOSE)) == 0) {
1376                                zinflag = TRUE;
1377                        } else if (strcmp(optarg,
1378                            MSG_ORIG(MSG_ARG_IGNORE)) == 0) {
1379                                ofl->ofl_flags1 |= FLG_OF1_IGNPRC;
1380                        } else if (strcmp(optarg,
1381                            MSG_ORIG(MSG_ARG_RELAXRELOC)) == 0) {
1382                                ofl->ofl_flags1 |= FLG_OF1_RLXREL;
1383                        } else if (strcmp(optarg,
1384                            MSG_ORIG(MSG_ARG_NORELAXRELOC)) == 0) {
1385                                ofl->ofl_flags1 |= FLG_OF1_NRLXREL;
1386                        } else if (strcmp(optarg,
1387                            MSG_ORIG(MSG_ARG_NOLDYNSYM)) == 0) {
1388                                ofl->ofl_flags |= FLG_OF_NOLDYNSYM;
1389                        } else if (strcmp(optarg,
1390                            MSG_ORIG(MSG_ARG_GLOBAUDIT)) == 0) {
1391                                ofl->ofl_dtflags_1 |= DF_1_GLOBAUDIT;
1392                        } else if (strcmp(optarg,
1393                            MSG_ORIG(MSG_ARG_NOSIGHANDLER)) == 0) {
1394                                ofl->ofl_flags1 |= FLG_OF1_NOSGHND;
1395                        } else if (strcmp(optarg,
1396                            MSG_ORIG(MSG_ARG_SYMBOLCAP)) == 0) {
1397                                ofl->ofl_flags |= FLG_OF_OTOSCAP;

1399                        /*
1400                         * Check archive group usage
1401                         *      -z rescan-start ... -z rescan-end
1402                         * to ensure they don't overlap and are well formed.
1403                         */
1404                        } else if (strcmp(optarg,
1405                            MSG_ORIG(MSG_ARG_RESCAN_START)) == 0) {
1406                                if (ofl->ofl_ars_gsandx == 0) {
1407                                        ofl->ofl_ars_gsandx = ndx;
1408                                } else if (ofl->ofl_ars_gsandx > 0) {
1409                                        /* Another group is still open */
1410                                        ld_eprintf(ofl, ERR_FATAL,
1411                                            MSG_INTL(MSG_ARG_AR_GRP_OLAP),
1412                                            MSG_INTL(MSG_MARG_AR_GRPS));
1413                                        /* Don't report cascading errors */
1414                                        ofl->ofl_ars_gsandx = -1;
1415                                }
1416                        } else if (strcmp(optarg,
1417                            MSG_ORIG(MSG_ARG_RESCAN_END)) == 0) {
1418                                if (ofl->ofl_ars_gsandx > 0) {
1419                                        ofl->ofl_ars_gsandx = 0;
```



```
1420                                } else if (ofl->ofl_ars_gsandx == 0) {
1421                                        /* There was no matching begin */
1422                                        ld_eprintf(ofl, ERR_FATAL,
1423                                            MSG_INTL(MSG_ARG_AR_GRP_BAD),
1424                                            MSG_INTL(MSG_MARG_AR_GRP_END),
1425                                            MSG_INTL(MSG_MARG_AR_GRP_START));
1426                                        /* Don't report cascading errors */
1427                                        ofl->ofl_ars_gsandx = -1;
1428                                }

1430                        /*
1431                         * If -z wrap is seen, enter the symbol to be wrapped
1432                         * into the wrap AVL tree.
1433                         */
1434                        } else if (strncmp(optarg, MSG_ORIG(MSG_ARG_WRAP),
1435                            MSG_ARG_WRAP_SIZE) == 0) {
1436                                if (ld_wrap_enter(ofl,
1437                                    optarg + MSG_ARG_WRAP_SIZE) == NULL)
1438                                        return (S_ERROR);
1439                        } else if (strncmp(optarg, MSG_ORIG(MSG_ARG_ASLR),
1440                            MSG_ARG_ASLR_SIZE) == 0) {
1441                                char *p = optarg + MSG_ARG_ASLR_SIZE;
1442                                if (*p == '\0') {
1443                                        ofl->ofl_aslr = 1;
1444                                } else if (*p == '=') {
1445                                        p++;

1447                                        if ((strcmp(p,
1448                                            MSG_ORIG(MSG_ARG_ENABLED)) == 0) ||
1449                                            (strcmp(p,
1450                                            MSG_ORIG(MSG_ARG_ENABLE)) == 0)) {
1447                                        if (strcmp(p,
1448                                            MSG_ORIG(MSG_ARG_ENABLED)) == 0) {
1451                                                ofl->ofl_aslr = 1;
1452                                        } else if ((strcmp(p,
1453                                            MSG_ORIG(MSG_ARG_DISABLED)) == 0) ||
1454                                            (strcmp(p,
1455                                            MSG_ORIG(MSG_ARG_DISABLE)) == 0)) {
1450                                        } else if (strcmp(p,
1451                                            MSG_ORIG(MSG_ARG_DISABLED)) == 0) {
1456                                                ofl->ofl_aslr = -1;
1457                                        } else {
1458                                                ld_eprintf(ofl, ERR_FATAL,
1459                                                    MSG_INTL(MSG_ARG_ILLEGAL),
1460                                                    MSG_ORIG(MSG_ARG_ZASLR), p);
1461                                                return (S_ERROR);
1462                                        }
1463                                } else {
1464                                        ld_eprintf(ofl, ERR_FATAL,
1465                                            MSG_INTL(MSG_ARG_ILLEGAL),
1466                                            MSG_ORIG(MSG_ARG_Z), optarg);
1467                                        return (S_ERROR);
1468                                }
1469                        } else if ((strncmp(optarg, MSG_ORIG(MSG_ARG_GUIDE),
1470                            MSG_ARG_GUIDE_SIZE) == 0) &&
1471                            ((optarg[MSG_ARG_GUIDE_SIZE] == '=') ||
1472                            (optarg[MSG_ARG_GUIDE_SIZE] == '\0'))) {
1473                                if (!guidance_parse(ofl, optarg))
1474                                        return (S_ERROR);
1475                        } else if (strcmp(optarg,
1476                            MSG_ORIG(MSG_ARG_FATWARN)) == 0) {
1477                                if (zfwflag  == SET_FALSE) {
1478                                        ld_eprintf(ofl, ERR_WARNING_NF,
1479                                            MSG_INTL(MSG_ARG_MTONCE),
1480                                            MSG_ORIG(MSG_ARG_ZFATWNOFATW));
1481                                } else {
```

```
1482                                    zfwflag = SET_TRUE;
1483                                    ofl->ofl_flags |= FLG_OF_FATWARN;
1484                            }
1485                    } else if (strcmp(optarg,
1486                        MSG_ORIG(MSG_ARG_NOFATWARN)) == 0) {
1487                            if (zfwflag  == SET_TRUE)
1488                                    ld_eprintf(ofl, ERR_WARNING_NF,
1489                                        MSG_INTL(MSG_ARG_MTONCE),
1490                                        MSG_ORIG(MSG_ARG_ZFATWNOFATW));
1491                            else
1492                                    zfwflag = SET_FALSE;

1494                    /*
1495                     * Process everything related to -z assert-deflib. This
1496                     * must be done in pass 1 because it gets used in pass
1497                     * 2.
1498                     */
1499                    } else if (strncmp(optarg, MSG_ORIG(MSG_ARG_ASSDEFLIB),
1500                        MSG_ARG_ASSDEFLIB_SIZE) == 0) {
1501                            if (assdeflib_parse(ofl, optarg) != TRUE)
1502                                    return (S_ERROR);
1503                    /*
1504                     * The following options just need validation as they
1505                     * are interpreted on the second pass through the
1506                     * command line arguments.
1507                     */
1508                    } else if (
1509                        strncmp(optarg, MSG_ORIG(MSG_ARG_INITARRAY),
1510                        MSG_ARG_INITARRAY_SIZE) &&
1511                        strncmp(optarg, MSG_ORIG(MSG_ARG_FINIARRAY),
1512                        MSG_ARG_FINIARRAY_SIZE) &&
1513                        strncmp(optarg, MSG_ORIG(MSG_ARG_PREINITARRAY),
1514                        MSG_ARG_PREINITARRAY_SIZE) &&
1515                        strncmp(optarg, MSG_ORIG(MSG_ARG_RTLDINFO),
1516                        MSG_ARG_RTLDINFO_SIZE) &&
1517                        strncmp(optarg, MSG_ORIG(MSG_ARG_DTRACE),
1518                        MSG_ARG_DTRACE_SIZE) &&
1519                        strcmp(optarg, MSG_ORIG(MSG_ARG_ALLEXTRT)) &&
1520                        strcmp(optarg, MSG_ORIG(MSG_ARG_DFLEXTRT)) &&
1521                        strcmp(optarg, MSG_ORIG(MSG_ARG_DIRECT)) &&
1522                        strcmp(optarg, MSG_ORIG(MSG_ARG_NODIRECT)) &&
1523                        strcmp(optarg, MSG_ORIG(MSG_ARG_GROUPPERM)) &&
1524                        strcmp(optarg, MSG_ORIG(MSG_ARG_LAZYLOAD)) &&
1525                        strcmp(optarg, MSG_ORIG(MSG_ARG_NOGROUPPERM)) &&
1526                        strcmp(optarg, MSG_ORIG(MSG_ARG_NOLAZYLOAD)) &&
1527                        strcmp(optarg, MSG_ORIG(MSG_ARG_NODEFERRED)) &&
1528                        strcmp(optarg, MSG_ORIG(MSG_ARG_RECORD)) &&
1529                        strcmp(optarg, MSG_ORIG(MSG_ARG_ALTEXEC64)) &&
1530                        strcmp(optarg, MSG_ORIG(MSG_ARG_WEAKEXT)) &&
1531                        strncmp(optarg, MSG_ORIG(MSG_ARG_TARGET),
1532                        MSG_ARG_TARGET_SIZE) &&
1533                        strcmp(optarg, MSG_ORIG(MSG_ARG_RESCAN_NOW)) &&
1534                        strcmp(optarg, MSG_ORIG(MSG_ARG_DEFERRED))) {
1535                            ld_eprintf(ofl, ERR_FATAL,
1536                                MSG_INTL(MSG_ARG_ILLEGAL),
1537                                MSG_ORIG(MSG_ARG_Z), optarg);
1538                    }

1540                    break;

1542            case 'D':
1543                    /*
1544                     * If we have not yet read any input files go ahead
1545                     * and process any debugging options (this allows any
1546                     * argument processing, entrance criteria and library
1547                     * initialization to be displayed).  Otherwise, if an
```



```
1548                     * input file has been seen, skip interpretation until
1549                     * process_files (this allows debugging to be turned
1550                     * on and off around individual groups of files).
1551                     */
1552                    Dflag = 1;
1553                    if (ofl->ofl_objscnt == 0) {
1554                            if (dbg_setup(ofl, optarg, 2) == 0)
1555                                    return (S_ERROR);
1556                    }

1558                    /*
1559                     * A diagnostic can only be provided after dbg_setup().
1560                     * As this is the first diagnostic that can be produced
1561                     * by ld(1), issue a title for timing and basic output.
1562                     */
1563                    if ((optitle == 0) && DBG_ENABLED) {
1564                            optitle++;
1565                            DBG_CALL(Dbg_basic_options(ofl->ofl_lml));
1566                    }
1567                    DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1568                    break;

1570            case 'B':
1571                    DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1572                    if (strcmp(optarg, MSG_ORIG(MSG_ARG_DIRECT)) == 0) {
1573                            if (Bdflag == SET_FALSE) {
1574                                    ld_eprintf(ofl, ERR_FATAL,
1575                                        MSG_INTL(MSG_ARG_INCOMP),
1576                                        MSG_ORIG(MSG_ARG_BNODIRECT),
1577                                        MSG_ORIG(MSG_ARG_BDIRECT));
1578                            } else {
1579                                    Bdflag = SET_TRUE;
1580                                    ofl->ofl_guideflags |= FLG_OFG_NO_DB;
1581                            }
1582                    } else if (strcmp(optarg,
1583                        MSG_ORIG(MSG_ARG_NODIRECT)) == 0) {
1584                            if (Bdflag == SET_TRUE) {
1585                                    ld_eprintf(ofl, ERR_FATAL,
1586                                        MSG_INTL(MSG_ARG_INCOMP),
1587                                        MSG_ORIG(MSG_ARG_BDIRECT),
1588                                        MSG_ORIG(MSG_ARG_BNODIRECT));
1589                            } else {
1590                                    Bdflag = SET_FALSE;
1591                                    ofl->ofl_guideflags |= FLG_OFG_NO_DB;
1592                            }
1593                    } else if (strcmp(optarg,
1594                        MSG_ORIG(MSG_STR_SYMBOLIC)) == 0)
1595                            Bsflag = TRUE;
1596                    else if (strcmp(optarg, MSG_ORIG(MSG_ARG_REDUCE)) == 0)
1597                            ofl->ofl_flags |= FLG_OF_PROCRED;
1598                    else if (strcmp(optarg, MSG_ORIG(MSG_STR_LOCAL)) == 0)
1599                            Blflag = TRUE;
1600                    else if (strcmp(optarg, MSG_ORIG(MSG_ARG_GROUP)) == 0)
1601                            Bgflag = TRUE;
1602                    else if (strcmp(optarg,
1603                        MSG_ORIG(MSG_STR_ELIMINATE)) == 0)
1604                            Beflag = TRUE;
1605                    else if (strcmp(optarg,
1606                        MSG_ORIG(MSG_ARG_TRANSLATOR)) == 0) {
1607                            ld_eprintf(ofl, ERR_WARNING,
1608                                MSG_INTL(MSG_ARG_UNSUPPORTED),
1609                                MSG_ORIG(MSG_ARG_BTRANSLATOR));
1610                    } else if (strcmp(optarg,
1611                        MSG_ORIG(MSG_STR_LD_DYNAMIC)) &&
1612                        strcmp(optarg, MSG_ORIG(MSG_ARG_STATIC))) {
1613                            ld_eprintf(ofl, ERR_FATAL,
```

```
1614                                    MSG_INTL(MSG_ARG_ILLEGAL),
1615                                    MSG_ORIG(MSG_ARG_CB), optarg);
1616                            }
1617                            break;

1619                    case 'G':
1620                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1621                            Gflag = TRUE;
1622                            break;

1624                    case 'L':
1625                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1626                            break;

1628                    case 'M':
1629                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1630                            if (aplist_append(&(ofl->ofl_maps), optarg,
1631                                AL_CNT_OFL_MAPFILES) == NULL)
1632                                    return (S_ERROR);
1633                            break;

1635                    case 'N':
1636                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1637                            break;

1639                    case 'Q':
1640                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1641                            if ((optarg[0] == 'n') && (optarg[1] == '\0')) {
1642                                    if (Qflag != SET_UNKNOWN)
1643                                            ld_eprintf(ofl, ERR_WARNING_NF,
1644                                                MSG_INTL(MSG_ARG_MTONCE),
1645                                                MSG_ORIG(MSG_ARG_CQ));
1646                                    else
1647                                            Qflag = SET_FALSE;
1648                            } else if ((optarg[0] == 'y') && (optarg[1] == '\0')) {
1649                                    if (Qflag != SET_UNKNOWN)
1650                                            ld_eprintf(ofl, ERR_WARNING_NF,
1651                                                MSG_INTL(MSG_ARG_MTONCE),
1652                                                MSG_ORIG(MSG_ARG_CQ));
1653                                    else
1654                                            Qflag = SET_TRUE;
1655                            } else {
1656                                    ld_eprintf(ofl, ERR_FATAL,
1657                                        MSG_INTL(MSG_ARG_ILLEGAL),
1658                                        MSG_ORIG(MSG_ARG_CQ), optarg);
1659                            }
1660                            break;

1662                    case 'S':
1663                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1664                            if (aplist_append(&lib_support, optarg,
1665                                AL_CNT_SUPPORT) == NULL)
1666                                    return (S_ERROR);
1667                            break;

1669                    case 'V':
1670                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1671                            if (!Vflag)
1672                                    (void) fprintf(stderr, MSG_ORIG(MSG_STR_STRNL),
1673                                        ofl->ofl_sgsid);
1674                            Vflag = TRUE;
1675                            break;

1677                    case 'Y':
1678                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, optarg));
1679                            if (strncmp(optarg, MSG_ORIG(MSG_ARG_LCOM), 2) == 0) {
```



```
1680                                    if (Llibdir)
1681                                            ld_eprintf(ofl, ERR_WARNING_NF,
1682                                                MSG_INTL(MSG_ARG_MTONCE),
1683                                                MSG_ORIG(MSG_ARG_CYL));
1684                                    else
1685                                            Llibdir = optarg + 2;
1686                            } else if (strncmp(optarg,
1687                                MSG_ORIG(MSG_ARG_UCOM), 2) == 0) {
1688                                    if (Ulibdir)
1689                                            ld_eprintf(ofl, ERR_WARNING_NF,
1690                                                MSG_INTL(MSG_ARG_MTONCE),
1691                                                MSG_ORIG(MSG_ARG_CYU));
1692                                    else
1693                                            Ulibdir = optarg + 2;
1694                            } else if (strncmp(optarg,
1695                                MSG_ORIG(MSG_ARG_PCOM), 2) == 0) {
1696                                    if (Plibpath)
1697                                            ld_eprintf(ofl, ERR_WARNING_NF,
1698                                                MSG_INTL(MSG_ARG_MTONCE),
1699                                                MSG_ORIG(MSG_ARG_CYP));
1700                                    else
1701                                            Plibpath = optarg + 2;
1702                            } else {
1703                                    ld_eprintf(ofl, ERR_FATAL,
1704                                        MSG_INTL(MSG_ARG_ILLEGAL),
1705                                        MSG_ORIG(MSG_ARG_CY), optarg);
1706                            }
1707                            break;

1709                    case '?':
1710                            DBG_CALL(Dbg_args_option(ofl->ofl_lml, ndx, c, NULL));
1711                            /*
1712                             * If the option character is '-', we're looking at a
1713                             * long option which couldn't be translated, display a
1714                             * more useful error.
1715                             */
1716                            if (optopt == '-') {
1717                                    eprintf(ofl->ofl_lml, ERR_FATAL,
1718                                        MSG_INTL(MSG_ARG_LONG_UNKNOWN),
1719                                        argv[optind-1]);
1720                            } else {
1721                                    eprintf(ofl->ofl_lml, ERR_FATAL,
1722                                        MSG_INTL(MSG_ARG_UNKNOWN), optopt);
1723                            }
1724                            (*usage)++;
1725                            break;

1727                    default:
1728                            break;
1729                    }

1731                    /*
1732                     * Update the argument index for the next getopt() iteration.
1733                     */
1734                    ndx = optind;
1735            }
1736            return (1);
1737 }
_____unchanged_portion_omitted_
```

```
**********************************************************
   60508 Fri Nov 18 16:39:49 2016
new/usr/src/cmd/sgs/libld/common/libld.msg
7594 ld -zaslr should accept Solaris-compatible values
**********************************************************
```

```
#
# 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) 1995, 2010, Oracle and/or its affiliates. All rights reserved.
#

#
# Copyright (c) 2012, Joyent, Inc.  All rights reserved.
#

@ _START_

# Message file for cmd/sgs/libld.

@ MSG_ID_LIBLD

#
# TRANSLATION_NOTE -- Beginning of USAGE message
#       The following messages are the usage messages for the ld command.
#       Tab characters (\t) are used to align the messages.
#
#       Each usage message starts with \t, and if the message has more than one
#               line, the following messages are aligned by 3 tab characters.
#               When you see \n\t\t\t, the first \n is used to change the line,
#               and following 3 tab characters are used to align the line.
#
#       Each usage message option is surrounded by [ and ].  Then the
#       description of the option follows. The descriptions should be aligned,
#       so tab characters are padded as needed after the closing bracket ].
#
#       How to align the messages are up to the translators and the
#       localization engineers.
#
#       In C locale, the first 3 messages would look like the following:
#
#       usage: ld [-6:abc:.....] file(s)
#               [-a]            create an absolute file
#               [-b]            do not do special PIC relocations in a.out
#               [-c file]       record configuration 'file'
#
@ MSG_ARG_USAGE         "usage: ld [-%s] file(s)\n"
@ MSG_ARG_DETAIL_3      "\t[-32]\t\tenforce a 32-bit link-edit\n"
```



```
@ MSG_ARG_DETAIL_6      "\t[-64]\t\tenforce a 64-bit link-edit\n"
@ MSG_ARG_DETAIL_A      "\t[-a]\t\tcreate an absolute file\n"
@ MSG_ARG_DETAIL_B      "\t[-b]\t\tdo not do special PIC relocations in a.out\n"
@ MSG_ARG_DETAIL_CBDR   "\t[-B direct | nodirect]\n\
                         \t\t\testablish direct bindings, or inhibit direct \
                         binding\n\
                         \t\t\tto, the object being created\n"
@ MSG_ARG_DETAIL_CBDY   "\t[-B dynamic | static]\n\
                         \t\t\tsearch for shared libraries|archives\n"
@ MSG_ARG_DETAIL_CBE    "\t[-B eliminate]\teliminate unqualified global \
                         symbols from the\n\t\t\tsymbol table\n"
@ MSG_ARG_DETAIL_CBG    "\t[-B group]\trelocate object from within group\n"
@ MSG_ARG_DETAIL_CBL    "\t[-B local]\treduce unqualified global symbols to \
                         local\n"
@ MSG_ARG_DETAIL_CBR    "\t[-B reduce]\tprocess symbol reductions\n"
@ MSG_ARG_DETAIL_CBS    "\t[-B symbolic]\tbind external references to \
                         definitions when creating\n\
                         \t\t\tshared objects\n"
@ MSG_ARG_DETAIL_C      "\t[-c name]\trecord configuration file 'name'\n"
@ MSG_ARG_DETAIL_CC     "\t[-C]\t\tdemangle C++ symbol name diagnostics\n"
@ MSG_ARG_DETAIL_D      "\t[-d y | n]\toperate in dynamic|static mode\n"
@ MSG_ARG_DETAIL_CD     "\t[-D token,...]\tprint diagnostic messages\n"
@ MSG_ARG_DETAIL_E      "\t[-e epsym], [--entry epsym]\n\
                         \t\t\tuse 'epsym' as entry point address\n"
@ MSG_ARG_DETAIL_F      "\t[-f name], [--auxiliary name]\n\
                         \t\t\tspecify library for which this file is an \
                         auxiliary\n\t\t\tfilter\n"
@ MSG_ARG_DETAIL_CF     "\t[-F name], [--filter name]\n\
                         \t\t\tspecify library for which this file is a filter\n
@ MSG_ARG_DETAIL_CG     "\t[-G], [-shared]\n\
                         \t\t\tcreate a shared object\n"
@ MSG_ARG_DETAIL_H      "\t[-h name], [--soname name]\n\
                         \t\t\tuse 'name' as internal shared object identifier\n
@ MSG_ARG_DETAIL_I      "\t[-i]\t\tignore LD_LIBRARY_PATH setting\n"
@ MSG_ARG_DETAIL_CI     "\t[-I name]\tuse 'name' as path of interpreter\n"
@ MSG_ARG_DETAIL_L      "\t[-l x], [--library x]\n\
                         \t\t\tsearch for libx.so or libx.a\n"
@ MSG_ARG_DETAIL_CL     "\t[-L path], [--library-path path]\n\
                         \t\t\tsearch for libraries in directory 'path'\n"
@ MSG_ARG_DETAIL_M      "\t[-m]\t\tprint memory map\n"
@ MSG_ARG_DETAIL_CM     "\t[-M mapfile]\tuse processing directives contained \
                         in 'mapfile'\n"
@ MSG_ARG_DETAIL_CN     "\t[-N string]\tcreate a dynamic dependency for \
                         'string'\n"
@ MSG_ARG_DETAIL_O      "\t[-o outfile], [--output outfile]\n\
                         \t\t\tname the output file 'outfile'\n"
@ MSG_ARG_DETAIL_P      "\t[-p auditlib]\tidentify audit library to accompany \
                         this object\n"
@ MSG_ARG_DETAIL_CP     "\t[-P auditlib]\tidentify audit library for \
                         processing the dependencies\n\
                         \t\t\tof this object\n"
@ MSG_ARG_DETAIL_CQ     "\t[-Q y | n]\tdo|do not place version information in \
                         output file\n"
@ MSG_ARG_DETAIL_R      "\t[-r], [--relocatable]\n\
                         \t\t\tcreate a relocatable object\n"
@ MSG_ARG_DETAIL_CR     "\t[-R path], [-rpath path]\n\
                         \t\t\tspecify a library search path to be used at run \
                         time\n"
@ MSG_ARG_DETAIL_S      "\t[-s], [--strip-all]\n\
                         \t\t\tstrip any symbol and debugging information\n"
@ MSG_ARG_DETAIL_CS     "\t[-S supportlib]\n\
                         \t\t\tspecify a link-edit support library\n"
@ MSG_ARG_DETAIL_T      "\t[-t]\t\tdo not warn of multiply-defined symbols \
                         that have\n\t\t\tdifferent sizes or alignments\n"
@ MSG_ARG_DETAIL_U      "\t[-u symname], [--undefined symname]\n\
                         \t\t\tcreate an undefined symbol 'symname'\n"
```

```
128 @ MSG_ARG_DETAIL_CV     "\t[-V], [--version]\n\
129                          \t\t\tprint version information\n"
130 @ MSG_ARG_DETAIL_CY     "\t[-Y P,dirlist]\tuse 'dirlist' as a default path \
131                          when searching for\n\
132                          \t\t\tlibraries\n"
133 @ MSG_ARG_DETAIL_ZA     "\t[-z absexec]\twhen building an executable absolute \
134                          symbols\n \
135                          \t\t\treferenced in dynamic objects are promoted to\n \
136                          \t\t\tthe executable\n"
137 @ MSG_ARG_DETAIL_ZAE    "\t[-z allextract | defaultextract | weakextract],\n\
138                          \t[--whole-archive | --no-whole-archive]\n\
139                          \t\t\textract all member files, only members that \
140                          resolve\n\
141                          \t\t\tundefined or tentative symbols, or \
142                          allow extraction of\n\
143                          \t\t\tarchive members to resolve weak references from \
144                          \n\t\t\t\tarchive files\n"
145 @ MSG_ARG_DETAIL_ZAL    "\t[-z altexec64]\texecute the 64-bit link-editor\n"
146 @ MSG_ARG_DETAIL_ZADLIB "\t[-z assert-deflib]\n\
147                         \t\t\tenables warnings for linking with libraries in \
148                         the \n\t\t\tdefault search path\n\
149                         \t[-z assert-deflib=libname]\n\
150                         \t\t\tenables warnings for linking with libraries in \
151                         the \n\t\t\tdefault search path, but 'libname' is exempt
152 @ MSG_ARG_DETAIL_ZC     "\t[-z combreloc | nocombreloc]\n\
153                          \t\t\tcombine|do not combine multiple relocation \
154                          sections\n"
155 @ MSG_ARG_DETAIL_ZNC    "\t[-z nocompstrtab]\n\t\t\tdisable compression of \
156                          string tables\n"
157 @ MSG_ARG_DETAIL_ZDEF   "\t[-z deferred | nodeferred]\n\
158                          \t\t\tenable|disable deferred identification of \
159                          shared object\n\t\t\tdependencies\n"
160 @ MSG_ARG_DETAIL_ZDFS   "\t[-z defs], [--no-undefined]\n\
161                          \t\t\tdisallow undefined symbol references\n"
162 @ MSG_ARG_DETAIL_ZDRS   "\t[-z direct | nodirect]\n\
163                          \t\t\tenable|disable direct binding to shared object\n\
164                          \t\t\tdependencies\n"
165 @ MSG_ARG_DETAIL_ZE     "\t[-z endfiltee]\tmarks a filtee such that it will \
166                          terminate a filters\n\t\t\tsearch\n"
167 @ MSG_ARG_DETAIL_ZFATW  "\t[-z fatal-warnings | nofatal-warnings],\n\
168                          \t[--fatal-warnings | --no-fatal-warnings]\n\
169                          \t\tenable|disable treatment of warnings as fatal\n"
170 @ MSG_ARG_DETAIL_ZFA    "\t[-z finiarray=function]\n\
171                          \t\t\tname of function to be appended to the \
172                          .fini_array\n"
173 @ MSG_ARG_DETAIL_ZGP    "\t[-z groupperm | nogroupperm]\n\
174                          \t\t\tenable|disable setting of group permissions\n\
175                          \t\t\ton dynamic dependencies\n"
176 @ MSG_ARG_DETAIL_ZGUIDE "\t[-z guidance | -z guidance=item1,item2,...]\n\
177                          \t\t\tenable guidance warnings. items: \
178                          noall, nodefs,\n\
179                          \t\t\tnodirect, nolazyload, nomapfile, notext, \
180                          nounused\n"
181 @ MSG_ARG_DETAIL_ZH     "\t[-z help], [--help]\n\
182                          \t\t\tprint this usage message\n"
183 @ MSG_ARG_DETAIL_ZIG    "\t[-z ignore | record]\n\
184                          \t\t\tignore|record unused dynamic dependencies\n"
185 @ MSG_ARG_DETAIL_ZINA   "\t[-z initarray=function]\n\
186                          \t\t\tname of function to be appended to the \
187                          .init_array\n"
188 @ MSG_ARG_DETAIL_ZINI   "\t[-z initfirst]\tmark object to indicate that its \
189                          .init section should\n\
190                          \t\t\tbe executed before the .init section of any \
191                          other\n\t\t\tobjects\n"
192 @ MSG_ARG_DETAIL_ZINT   "\t[-z interpose]\
193                          \tdynamic object is to be an 'interposer' on direct\n\
```



```
194                          \t\t\tbindings\n"
195 @ MSG_ARG_DETAIL_ZLAZY  "\t[-z lazyload | nolazyload]\n\
196                          \t\t\tenable|disable delayed loading of shared \
197                          object\n\t\t\tdependencies\n"
198 @ MSG_ARG_DETAIL_ZLD32  "\t[-z ld32=arg1,arg2,...]\n\
199                          \t\t\tdefine arguments applicable to the \
200                          32-bit class of ld(1)\n"
201 @ MSG_ARG_DETAIL_ZLD64  "\t[-z ld64=arg1,arg2,...]\n\
202                          \t\t\tdefine arguments applicable to the \
203                          64-bit class of ld(1)\n"
204 @ MSG_ARG_DETAIL_ZLO    "\t[-z loadfltr]\tmark filter as requiring immediate \
205                          loading of its\n\
206                          \t\t\tfiltees at runtime\n"
207 @ MSG_ARG_DETAIL_ZM     "\t[-z muldefs], [--allow-multiple-definition]\n\
208                          \t\t\tallow multiply-defined symbols\n"
209 @ MSG_ARG_DETAIL_ZNDFS  "\t[-z nodefs]\tallow undefined symbol references\n"
210 @ MSG_ARG_DETAIL_ZNDEF  "\t[-z nodefaultlib]\n\
211                          \t\t\tmark object to ignore any default library \
212                          search path\n"
213 @ MSG_ARG_DETAIL_ZNDEL  "\t[-z nodelete]\tmark object as non-deletable\n"
214 @ MSG_ARG_DETAIL_ZNDLO  "\t[-z nodlopen]\tmark object as non-dlopen()'able\n"
215 @ MSG_ARG_DETAIL_ZNDU   "\t[-z nodump]\tmark object as non-dldump()'able\n"
216 @ MSG_ARG_DETAIL_ZNLD   "\t[-z noldynsym]\tdo not add a .SUNW_ldynsym section\n"
217 @ MSG_ARG_DETAIL_ZNPA   "\t[-z nopartial]\texpand any partially initialized \
218                          symbols\n"
219 @ MSG_ARG_DETAIL_ZNV    "\t[-z noversion]\tdo not record any version sections\n"
220 @ MSG_ARG_DETAIL_ZNOW   "\t[-z now]\tmark object as requiring non-lazy \
221                          binding\n"
222 @ MSG_ARG_DETAIL_ZO     "\t[-z origin]\tmark object as requiring $ORIGIN \
223                          processing\n"
224 @ MSG_ARG_DETAIL_ZPIA   "\t[-z preinitarray=function]\n\
225                          \t\tname of function to be appended to the \
226                          .preinit_array\n"
227 @ MSG_ARG_DETAIL_ZRL    "\t[-z redlocsym]\treduce local syms in .symtab to \
228                          a minimum\n"
229 @ MSG_ARG_DETAIL_ZRREL  "\t[-z relaxreloc]\trelax rules used for relocations \
230                          against COMDAT sections\n"
231 @ MSG_ARG_DETAIL_ZRS    "\t[-z rescan]\tafter processing all arguments, rescan \
232                          archive list\n\
233                          \t\tuntil no further member extraction occurs\n"
234 @ MSG_ARG_DETAIL_ZRSN   "\t[-z rescan-now]\timmediately rescan archive list \
235                          until\n\
236                          \t\tno further member extraction occurs\n"
237 @ MSG_ARG_DETAIL_ZRSGRP "\t[-z rescan-start archives... -z rescan-end],\n\
238                          \t[--start-group archives... --end-group], \
239                          [-( archives... -)]\n\
240                          \t\trescan specified archive group upon reaching\n\
241                          \t\tthe end of the group, until no further\n\
242                          \t\tmember extraction occurs\n"
243 @ MSG_ARG_DETAIL_ZSCAP  "\t[-z symbolcap]\tconvert object capabilities to \
244                          symbol capabilities\n"
245 @ MSG_ARG_DETAIL_ZTARG  "\t[-z target=platform]\n\
246                          \t\ttarget machine for cross linking\n"
247 @ MSG_ARG_DETAIL_ZT     "\t[-z text]\tdisallow output relocations against \
248                          text\n"
249 @ MSG_ARG_DETAIL_ZTO    "\t[-z textoff]\tallow output relocations against \
250                          text\n"
251 @ MSG_ARG_DETAIL_ZTW    "\t[-z textwarn]\twarn if there are relocations \
252                          against text\n"
253 @ MSG_ARG_DETAIL_ZWRAP  "\t[-z wrap=symbol], [-wrap=symbol], [--wrap=symbol]\n\
254                          \t\twrap symbol references\n"
255 @ MSG_ARG_DETAIL_ZVER   "\t[-z verbose]\t\
256                          generate warnings for suspicious processings\n"

258 #
259 # TRANSLATION_NOTE -- End of USAGE message
```

```
260 #
261 @ MSG_GRP_INVALNDX      "file %s: group section [%u]%s: entry %d: \
262                          invalid section index: %d"

264 # Relocation processing messages (some of these are required to satisfy
265 # do_reloc(), which is common code used by cmd/sgs/rtld - make sure both
266 # message files remain consistent).

268 @ MSG_REL_NOFIT         "relocation error: %s: file %s: symbol %s: \
269                          value 0x%llx does not fit"
270 @ MSG_REL_NONALIGN      "relocation error: %s: file %s: symbol %s: \
271                          offset 0x%llx is non-aligned"
272 @ MSG_REL_NULL          "relocation error: file %s: section [%u]%s: \
273                          skipping null relocation record"
274 @ MSG_REL_NOTSUP        "relocation error: %s: file %s: section [%u]%s: \
275                          relocation not currently supported"
276 @ MSG_REL_PICREDLOC     "relocation error: %s: file %s symbol %s: \
277                          -z redlocsym may not be used for pic code"
278 @ MSG_REL_TLSLE         "relocation error: %s: file %s: symbol %s: \
279                          relocation illegal when building a shared object"
280 @ MSG_REL_TLSBND        "relocation error: %s: file %s: symbol %s: \
281                          bound to: %s: relocation illegal when not bound \
282                          to object being created"
283 @ MSG_REL_TLSSTAT       "relocation error: %s: file %s: symbol %s: \
284                          relocation illegal when building a static object"
285 @ MSG_REL_TLSBADSYM     "relocation error: %s: file %s: symbol %s: \
286                          bad symbol type %s: symbol type must be TLS"
287 @ MSG_REL_BADTLS        "relocation error: %s: file %s: symbol %s: \
288                          relocation illegal for TLS symbol"
289 @ MSG_REL_BADGOTBASED   "relocation error: %s: file %s: symbol %s: a GOT \
290                          relative relocation must reference a local symbol"
291 @ MSG_REL_UNKNWSYM      "relocation error: %s: file %s: section [%u]%s: \
292                          attempt to relocate with respect to unknown \
293                          symbol %s: offset 0x%llx, symbol index %d"
294 @ MSG_REL_UNSUPSZ       "relocation error: %s: file %s: symbol %s: \
295                          offset size (%d bytes) is not supported"
296 @ MSG_REL_INVALOFFSET   "relocation error: %s: file %s section [%u]%s: \
297                          invalid offset symbol '%s': offset 0x%llx"
298 @ MSG_REL_INVALRELT     "relocation error: file %s: section [%u]%s: \
299                          invalid relocation type: 0x%x"
300 @ MSG_REL_EMPTYSEC      "relocation error: %s: file %s: symbol %s: \
301                          attempted against empty section [%u]%s"
302 @ MSG_REL_EXTERNSYM     "relocation error: %s: file %s: symbol %s: \
303                          external symbolic relocation against non-allocatable \
304                          section %s; cannot be processed at runtime: \
305                          relocation ignored"
306 @ MSG_REL_UNEXPREL      "relocation error: %s: file %s: symbol %s: \
307                          unexpected relocation; generic processing performed"
308 @ MSG_REL_UNEXPSYM      "relocation error: %s: file %s: symbol %s: \
309                          unexpected symbol referenced from file %s"
310 @ MSG_REL_SYMDISC       "relocation error: %s: file %s: section [%u]%s: \
311                          symbol %s: symbol has been discarded with discarded \
312                          section: [%u]%s"
313 @ MSG_REL_NOSYMBOL      "relocation error: %s: file %s: section: [%u]%s: \
314                          offset: 0x%llx: relocation requires reference symbol"
315 @ MSG_REL_DISPREL1      "relocation error: %s: file %s: symbol %s: \
316                          displacement relocation applied to the symbol \
317                          %s at 0x%llx: symbol %s is a copy relocated symbol"
318 @ MSG_REL_UNSUPSIZE     "relocation error: %s: file %s: section [%u]%s: \
319                          relocation against section symbol unsupported"

321 @ MSG_REL_DISPREL2      "relocation warning: %s: file %s: symbol %s: \
322                          may contain displacement relocation"
323 @ MSG_REL_DISPREL3      "relocation warning: %s: file %s: symbol %s: \
324                          displacement relocation applied to the symbol \
325                          %s: at 0x%llx: displacement relocation will not be \
```



```
326                          visible in output image"
327 @ MSG_REL_DISPREL4      "relocation warning: %s: file %s: symbol %s: \
328                          displacement relocation to be applied to the symbol \
329                          %s: at 0x%llx: displacement relocation will be \
330                          visible in output image"
331 @ MSG_REL_COPY          "relocation warning: %s: file %s: symbol %s: \
332                          relocation bound to a symbol with STV_PROTECTED \
333                          visibility"
334 @ MSG_RELINVSEC         "relocation warning: %s: file %s: section: [%u]%s: \
335                          against suspicious section [%u]%s; relocation ignored"
336 @ MSG_REL_TLSIE         "relocation warning: %s: file %s: symbol %s: \
337                          relocation has restricted use when building a shared \
338                          object"

340 @ MSG_REL_SLOPCDATNONAM "relocation warning: %s: file %s: section [%u]%s: \
341                          relocation against discarded COMDAT section [%u]%s: \
342                          redirected to file %s"
343 @ MSG_REL_SLOPCDATNAM   "relocation warning: %s: file %s: section [%u]%s: \
344                          symbol %s: relocation against discarded COMDAT \
345                          section [%u]%s: redirected to file %s"
346 @ MSG_REL_SLOPCDATNOSYM "relocation warning: %s: file %s: section [%u]%s: \
347                          symbol %s: relocation against discarded COMDAT \
348                          section [%u]%s: symbol not found, relocation ignored"

350 @ MSG_REL_NOREG         "relocation error: REGISTER relocation not supported \
351                          on target architecture"

353 #
354 # TRANSLATION_NOTE
355 #       The following 7 messages are the message to print the
356 #       following example messages.
357 #
358 #Text relocation remains                         referenced
359 #    against symbol                offset      in file
360 #str                                0x14        main.o
361 #printf                             0x1c        main.o
362 #
363 #       The first two lines are the header, and the next msgid
364 #       is the format string for the header.
365 #       Tabs and spaces are used for alignment.
366 #       The first and third %s are for: "Text relocation remains against symbol"
367 #       The second %s and fourth %s are for: "referenced in file"
368 #       The third %s is for: "offset"
369 #
370 @ MSG_REL_REMAIN_FMT_1  "%-40s\t%s\n    %s\t\t    %s\t%s"
371 #
372 # TRANSLATION_NOTE
373 #       The next two msdid make a sentence. So translate:
374 #               "Text relocation remain against symbol"
375 #       And separate them into two msgstr considering the proper
376 #       alignment.
377 @ MSG_REL_RMN_ITM_11    "Text relocation remains"
378 @ MSG_REL_RMN_ITM_12    "against symbol"
379 @ MSG_REL_RMN_ITM_13    "warning: Text relocation remains"

381 @ MSG_REL_RMN_ITM_2     "offset"

383 #
384 # TRANSLATION_NOTE
385 #       The next two msdid make a sentence. So translate:
386 #               "referenced in file"
387 #       And separate them into two msgstr considering the proper
388 #       alignment.
389 @ MSG_REL_RMN_ITM_31    "referenced"
390 @ MSG_REL_RMN_ITM_32    "in file"
391 @ MSG_REL_REMAIN_2      "%-35s 0x%-8llx\t%s"
```

```
392 @ MSG_REL_REMAIN_3      "relocations remain against allocatable but \
393                          non-writable sections"

395 # Files processing messages

397 @ MSG_FIL_MULINC_1      "file %s: attempted multiple inclusion of file"
398 @ MSG_FIL_MULINC_2      "file %s: linked to %s: attempted multiple inclusion \
399                          of file"
400 @ MSG_FIL_SOINSTAT      "input of shared object '%s' in static mode"
401 @ MSG_FIL_INVALSEC      "file %s: section [%u]%s has invalid type %s"
402 @ MSG_FIL_NOTFOUND      "file %s: required by %s, not found"
403 @ MSG_FIL_MALSTR        "file %s: section [%u]%s: malformed string table, \
404                          initial or final byte"
405 @ MSG_FIL_PTHTOLONG     "'%s/%s' pathname too long"
406 @ MSG_FIL_EXCLUDE       "file %s: section [%u]%s contains both SHF_EXCLUDE and \
407                          SHF_ALLOC flags: SHF_EXCLUDE ignored"
408 @ MSG_FIL_INTERRUPT     "file %s: creation interrupted: %s"
409 @ MSG_FIL_INVRELOC1     "file %s: section [%u]%s: relocations can not be \
410                          applied against section [%u]%s"
411 @ MSG_FIL_INVSHINFO     "file %s: section [%u]%s: has invalid sh_info: %lld"
412 @ MSG_FIL_INVSHLINK     "file %s: section [%u]%s: has invalid sh_link: %lld"
413 @ MSG_FIL_INVSHENTSIZE  "file %s: section [%u]%s: has invalid sh_entsize: %lld"
414 @ MSG_FIL_NOSTRTABLE    "file %s: section [%u]%s: symbol[%d]: specifies string \
415                          table offset 0x%llx: no string table is available"
416 @ MSG_FIL_EXCSTRTABLE   "file %s: section [%u]%s: symbol[%d]: specifies string \
417                          table offset 0x%llx: exceeds string table %s: \
418                          size 0x%llx"
419 @ MSG_FIL_NONAMESYM     "file %s: section [%u]%s: symbol[%d]: global symbol has
420                          no name"
421 @ MSG_FIL_UNKCAP        "file %s: section [%u]%s: unknown capability tag: %d"
422 @ MSG_FIL_BADSF1        "file %s: section [%u]%s: unknown software \
423                          capabilities: 0x%llx; ignored"
424 @ MSG_FIL_INADDR32SF1   "file %s: section [%u]%s: software capability ADDR32: is
425                          ineffective when building 32-bit object; ignored"
426 @ MSG_FIL_EXADDR32SF1   "file %s: section [%u]%s: software capability ADDR32: \
427                          requires executable be built with ADDR32 capability"

429 @ MSG_FIL_BADORDREF     "file %s: section [%u]%s: contains illegal reference \
430                          to discarded section: [%u]%s"

432 # Recording name conflicts

434 @ MSG_REC_OPTCNFLT      "recording name conflict: file '%s' and %s provide \
435                          identical dependency names: %s"
436 @ MSG_REC_OBJCNFLT      "recording name conflict: file '%s' and file '%s' \
437                          provide identical dependency names: %s  %s"
438 @ MSG_REC_CNFLTHINT     "(possible multiple inclusion of the same file)"

440 # System call messages

442 @ MSG_SYS_OPEN          "file %s: open failed: %s"
443 @ MSG_SYS_UNLINK        "file %s: unlink failed: %s"
444 @ MSG_SYS_MMAPANON      "mmap anon failed: %s"
445 @ MSG_SYS_MALLOC        "malloc failed: %s"


448 # Messages related to platform support

450 @ MSG_TARG_UNSUPPORTED  "unsupported ELF machine type: %s"


453 # ELF processing messages

455 @ MSG_ELF_LIBELF        "libelf: version not supported: %d"

457 @ MSG_ELF_ARMEM         "file %s: unable to locate archive member;\n\t\
```



```
458                          offset=%x, symbol=%s"

460 @ MSG_ELF_ARSYM         "file %s ignored: unable to locate archive symbol table"

462 @ MSG_ELF_VERSYM        "file %s: version symbol section entry mismatch:\n\t\
463                          (section [%u]%s entries=%d; section [%u]%s entries=%d)"

465 @ MSG_ELF_NOGROUPSECT   "file %s: section [%u]%s: SHF_GROUP flag set, but no \
466                          corresponding SHT_GROUP section found"

468 # Section processing errors

470 @ MSG_SCN_NONALLOC      "%s: non-allocatable section '%s' directed to a \
471                          loadable segment: %s"

473 @ MSG_SCN_MULTICOMDAT   "file %s: section [%u]%s: cannot be susceptible to multi
474                          COMDAT mechanisms: %s"

476 @ MSG_SCN_DWFOVRFLW     "%s: section %s: encoded DWARF data exceeds \
477                          section size"
478 @ MSG_SCN_DWFBADENC     "%s: section %s: invalid DWARF encoding: %#x"

480 # Symbol processing errors

482 @ MSG_SYM_NOSECDEF      "symbol '%s' in file %s has no section definition"
483 @ MSG_SYM_INVSEC        "symbol '%s' in file %s associated with invalid \
484                          section[%lld]"
485 @ MSG_SYM_TLS           "symbol '%s' in file %s (STT_TLS), is defined \
486                          in a non-SHF_TLS section"
487 @ MSG_SYM_BADADDR       "symbol '%s' in file %s: section [%u]%s: size %#llx: \
488                          symbol (address %#llx, size %#llx) lies outside \
489                          of containing section"
490 @ MSG_SYM_BADADDR_ROTXT "symbol '%s' in file %s: readonly text section \
491                          [%u]%s: size %#llx: symbol (address %#llx, \
492                          size %#llx) lies outside of containing section"
493 @ MSG_SYM_MULDEF        "symbol '%s' is multiply-defined:"
494 @ MSG_SYM_CONFVIS       "symbol '%s' has conflicting visibilities:"
495 @ MSG_SYM_DIFFTYPE      "symbol '%s' has differing types:"
496 @ MSG_SYM_DIFFATTR      "symbol '%s' has differing %s:\n\
497                          \t(file %s value=0x%llx; file %s value=0x%llx);"
498 @ MSG_SYM_FILETYPES     "\t(file %s type=%s; file %s type=%s);"
499 @ MSG_SYM_VISTYPES      "\t(file %s visibility=%s; file %s visibility=%s);"
500 @ MSG_SYM_DEFTAKEN      "\t%s definition taken"
501 @ MSG_SYM_DEFUPDATE     "\t%s definition taken and updated with larger size"
502 @ MSG_SYM_LARGER        "\tlargest value applied"
503 @ MSG_SYM_TENTERR       "\ttentative symbol cannot override defined symbol \
504                          of smaller size"

506 @ MSG_SYM_INVSHNDX      "symbol %s has invalid section index; \
507                          ignored:\n\t(file %s value=%s);"
508 @ MSG_SYM_NONGLOB       "global symbol %s has non-global binding:\n\
509                          \t(file %s value=%s);"
510 @ MSG_SYM_RESERVE       "reserved symbol '%s' already defined in file %s"
511 @ MSG_SYM_NOTNULL       "undefined symbol '%s' with non-zero value encountered \
512                          from file %s"
513 @ MSG_SYM_DUPSORTADDR   "section %s: symbol '%s' and symbol '%s' have the \
514                          same address: %#llx: remove duplicate with \
515                          NOSORTSYM mapfile directive"

517 @ MSG_PSYM_INVMINFO1    "file %s: section [%u]%s: entry[%d] has invalid m_info:
518                          0x%llx for symbol index"
519 @ MSG_PSYM_INVMINFO2    "file %s: section [%u]%s: entry[%d] has invalid m_info:
520                          0x%llx for size"
521 @ MSG_PSYM_INVMREPEAT   "file %s: section [%u]%s: entry[%d] has invalid m_repeat
522                          0x%llx"
523 @ MSG_PSYM_CANNOTEXPND  "file %s: section [%u]%s: entry[%d] can not be expanded:
```

```
524                          associated symbol size is unknown %s"
525 @ MSG_PSYM_NOSTATIC     "and partial initialization cannot be deferred to \
526                          a static object"
527 @ MSG_MOVE_OVERLAP      "file %s: section [%u]%s: symbol '%s' overlapping move \
528                          initialization: start=0x%llx, length=0x%llx: \
529                          start=0x%llx, length=0x%llx"
530 @ MSG_PSYM_EXPREASON1   "output file is static object"
531 @ MSG_PSYM_EXPREASON2   "-z nopartial option in effect"
532 @ MSG_PSYM_EXPREASON3   "move infrastructure size is greater than move data"

534 #
535 # Support library failures
536 #
537 @ MSG_SUP_NOLOAD        "dlopen() of support library (%s) failed with \
538                          error: %s"
539 @ MSG_SUP_BADVERSION    "initialization of support library (%s) failed with \
540                          bad version.  supported: %d returned: %d"


543 #
544 # TRANSLATION_NOTE
545 #       The following 7 messages are the message to print the
546 #       following example messages.
547 #
548 #Undefined                       first referenced
549 # symbol                            in file
550 #inquire                           halt_hold.o
551 #
552 @ MSG_SYM_FMT_UNDEF     "%s\t\t\t%s\
553                          \n %s  \t\t\t    %s"

555 #
556 # TRANSLATION_NOTE
557 #       The next two msdid make a sentence. So translate:
558 #               "Undefined symbol"
559 #       And separate them into two msgstr considering the proper
560 #       alignment.
561 @ MSG_SYM_UNDEF_ITM_11  "Undefined"
562 @ MSG_SYM_UNDEF_ITM_12  "symbol"
563 #
564 # TRANSLATION_NOTE
565 #       The next two msdid make a sentence. So translate:
566 #               "first referenced in file"
567 #       And separate them into two msgstr considering the proper
568 #       alignment.
569 @ MSG_SYM_UNDEF_ITM_21  "first referenced"
570 @ MSG_SYM_UNDEF_ITM_22  "in file"
571 #

573 @ MSG_SYM_UND_UNDEF     "%-35s %s"
574 @ MSG_SYM_UND_NOVER     "%-35s %s  (symbol has no version assigned)"
575 @ MSG_SYM_UND_IMPL      "%-35s %s  (symbol belongs to implicit dependency %s)"
576 @ MSG_SYM_UND_NOTA      "%-35s %s  (symbol belongs to unavailable version %s \
577                          (%s))"
578 @ MSG_SYM_UND_BNDLOCAL  "%-35s %s  (symbol scope specifies local binding)"

580 @ MSG_SYM_ENTRY         "entry point"
581 @ MSG_SYM_UNDEF         "%s symbol '%s' is undefined"
582 @ MSG_SYM_EXTERN        "%s symbol '%s' is undefined  (symbol belongs to \
583                          dependency %s)"
584 @ MSG_SYM_NOCRT         "symbol '%s' not found, but %s section exists - \
585                          possible link-edit without using the compiler driver"

587 # Output file update messages

589 @ MSG_UPD_NOREADSEG     "No read-only segments found.  Setting '_etext' to 0"
```



```
590 @ MSG_UPD_NORDWRSEG     "No read-write segments found.  Setting '_edata' to 0"
591 @ MSG_UPD_NOSEG         "Setting 'end' and '_end' to 0"

593 @ MSG_UPD_SEGOVERLAP    "%s: segment address overlap;\n\
594                          \tprevious segment ending at address 0x%llx overlaps\n\
595                          \tuser defined segment '%s' starting at address 0x%llx"
596 @ MSG_UPD_LARGSIZE      "%s: segment %s calculated size 0x%llx\n\
597                          \tis larger than user-defined size 0x%llx"

599 @ MSG_UPD_NOBITS        "NOBITS section found before end of initialized data"
600 @ MSG_SEG_FIRNOTLOAD    "First segment has type %s, PT_LOAD required: %s"
601 @ MSG_UPD_MULEHFRAME    "file %s; section [%u]%s and file %s; section [%u]%s \
602                          have incompatibile attributes and cannot \
603                          be merged into a single output section"


606 # Version processing messages

608 @ MSG_VER_HIGHER        "file %s: version revision %d is higher than \
609                          expected %d"
610 @ MSG_VER_NOEXIST       "file %s: version '%s' does not exist:\n\
611                          \trequired by file %s"
612 @ MSG_VER_UNDEF         "version '%s' undefined, referenced by version '%s':\n\
613                          \trequired by file %s"
614 @ MSG_VER_UNAVAIL       "file %s: version '%s' is unavailable:\n\
615                          \trequired by file %s"
616 @ MSG_VER_DEFINED       "version symbol '%s' already defined in file %s"
617 @ MSG_VER_INVALNDX      "version symbol '%s' from file %s has an invalid \
618                          version index (%d)"
619 @ MSG_VER_ADDVERS       "unused $ADDVERS specification from file '%s' \
620                          for object '%s'\nversion(s):"
621 @ MSG_VER_ADDVER        "\t%s"
622 @ MSG_VER_CYCLIC        "following versions generate cyclic dependency:"

624 # Capabilities messages

626 @ MSG_CAP_MULDEF        "capabilities symbol '%s' has multiply-defined members:"
627 @ MSG_CAP_MULDEFSYMS    "\t(file %s symbol '%s'; file %s symbol '%s');"
628 @ MSG_CAP_REDUNDANT     "file %s: section [%u]%s: symbol capabilities \
629                          redundant, as object capabilities are more restrictive"
630 @ MSG_CAP_NOSYMSFOUND   "no global symbols have been found that are associated \
631                          with capabilities identified relocatable objects: \
632                          -z symbolcap has no effect"

634 @ MSG_CAPINFO_INVALSYM  "file %s: capabilities info section [%u]%s: index %d: \
635                          family member symbol '%s': invalid"
636 @ MSG_CAPINFO_INVALLEAD "file %s: capabilities info section [%u]%s: index %d: \
637                          family lead symbol '%s': invalid symbol index %d"

639 # Basic strings

641 @ MSG_STR_ALIGNMENTS    "alignments"
642 @ MSG_STR_COMMAND       "(command line)"
643 @ MSG_STR_TLSREL        "(internal TLS relocation requirement)"
644 @ MSG_STR_SIZES         "sizes"
645 @ MSG_STR_UNKNOWN       "<unknown>"
646 @ MSG_STR_SECTION       "%s (section)"
647 @ MSG_STR_SECTION_MSTR  "%s (merged string section)"

649 #
650 # TRANSLATION_NOTE
651 #       The elf_ function name represents a man page reference and should not
652 #       be translated.
653 @ MSG_ELF_BEGIN         "file %s: elf_begin"
654 @ MSG_ELF_CNTL          "file %s: elf_cntl"
655 @ MSG_ELF_GETARHDR      "file %s: elf_getarhdr"
```

```
656 @ MSG_ELF_GETARSYM      "file %s: elf_getarsym"
657 @ MSG_ELF_GETDATA       "file %s: elf_getdata"
658 @ MSG_ELF_GETEHDR       "file %s: elf_getehdr"
659 @ MSG_ELF_GETPHDR       "file %s: elf_getphdr"
660 @ MSG_ELF_GETSCN        "file %s: elf_getscn: scnndx: %d"
661 @ MSG_ELF_GETSHDR       "file %s: elf_getshdr"
662 @ MSG_ELF_MEMORY        "file %s: elf_memory"
663 @ MSG_ELF_NDXSCN        "file %s: elf_ndxscn"
664 @ MSG_ELF_NEWDATA       "file %s: elf_newdata"
665 @ MSG_ELF_NEWEHDR       "file %s: elf_newehdr"
666 @ MSG_ELF_NEWSCN        "file %s: elf_newscn"
667 @ MSG_ELF_NEWPHDR       "file %s: elf_newphdr"
668 @ MSG_ELF_STRPTR        "file %s: elf_strptr"
669 @ MSG_ELF_UPDATE        "file %s: elf_update"
670 @ MSG_ELF_SWAP_WRIMAGE  "file %s: _elf_swap_wrimage"


673 @ MSG_REJ_MACH          "file %s: wrong ELF machine type: %s"
674 @ MSG_REJ_CLASS         "file %s: wrong ELF class: %s"
675 @ MSG_REJ_DATA          "file %s: wrong ELF data format: %s"
676 @ MSG_REJ_TYPE          "file %s: bad ELF type: %s"
677 @ MSG_REJ_BADFLAG       "file %s: bad ELF flags value: %s"
678 @ MSG_REJ_MISFLAG       "file %s: mismatched ELF flags value: %s"
679 @ MSG_REJ_VERSION       "file %s: mismatched ELF/lib version: %s"
680 @ MSG_REJ_HAL           "file %s: HAL R1 extensions required"
681 @ MSG_REJ_US3           "file %s: Sun UltraSPARC III extensions required"
682 @ MSG_REJ_STR           "file %s: %s"
683 @ MSG_REJ_UNKFILE       "file %s: unknown file type"
684 @ MSG_REJ_UNKCAP        "file=%s; unknown capability: %d"
685 @ MSG_REJ_HWCAP_1       "file %s: hardware capability (CA_SUNW_HW_1) \
686                          unsupported: %s"
687 @ MSG_REJ_SFCAP_1       "file %s: software capability (CA_SUNW_SF_1) \
688                          unsupported: %s"
689 @ MSG_REJ_MACHCAP       "file %s: machine capability (CA_SUNW_MACH) \
690                          unsupported: %s"
691 @ MSG_REJ_PLATCAP       "file %s: platform capability (CA_SUNW_PLAT) \
692                          unsupported: %s"
693 @ MSG_REJ_HWCAP_2       "file %s: hardware capability (CA_SUNW_HW_2) \
694                          unsupported: %s"
695 @ MSG_REJ_ARCHIVE       "file %s: invalid archive use"

697 # Guidance messages
698 @ MSG_GUIDE_SUMMARY     "see ld(1) -z guidance for more information"
699 @ MSG_GUIDE_DEFS        "-z defs option recommended for shared objects"
700 @ MSG_GUIDE_DIRECT      "-B direct or -z direct option recommended before \
701                          first dependency"
702 @ MSG_GUIDE_LAZYLOAD    "-z lazyload option recommended before \
703                          first dependency"
704 @ MSG_GUIDE_MAPFILE     "version 2 mapfile syntax recommended: %s"
705 @ MSG_GUIDE_TEXT        "position independent (PIC) code recommended for \
706                          shared objects"
707 @ MSG_GUIDE_UNUSED      "removal of unused dependency recommended: %s"

709 @ _END_


712 # The following strings represent reserved names.  Reference to these strings
713 # is via the MSG_ORIG() macro, and thus translations are not required.

715 @ MSG_STR_EOF           "<eof>"
716 @ MSG_STR_ERROR         "<error>"
717 @ MSG_STR_EMPTY         ""
718 @ MSG_QSTR_BANG         "'!'"
719 @ MSG_STR_COLON         ":"
720 @ MSG_QSTR_COLON        "':'"
721 @ MSG_QSTR_SEMICOLON    "';'"
```



```
722 @ MSG_QSTR_EQUAL        "'='"
723 @ MSG_QSTR_PLUSEQ       "'+='"
724 @ MSG_QSTR_MINUSEQ      "'-='"
725 @ MSG_QSTR_ATSIGN       "'@'"
726 @ MSG_QSTR_DASH         "'-'"
727 @ MSG_QSTR_LEFTBKT      "'{'"
728 @ MSG_QSTR_RIGHTBKT     "'}'"
729 @ MSG_QSTR_PIPE         "'|'"
730 @ MSG_QSTR_STAR         "'*'"
731 @ MSG_STR_DOT           "."
732 @ MSG_STR_SLASH         "/"
733 @ MSG_STR_DYNAMIC       "(.dynamic)"
734 @ MSG_STR_ORIGIN        "$ORIGIN"
735 @ MSG_STR_MACHINE       "$MACHINE"
736 @ MSG_STR_PLATFORM      "$PLATFORM"
737 @ MSG_STR_ISALIST       "$ISALIST"
738 @ MSG_STR_OSNAME        "$OSNAME"
739 @ MSG_STR_OSREL         "$OSREL"
740 @ MSG_STR_UU_REAL_U     "__real_"
741 @ MSG_STR_UU_WRAP_U     "__wrap_"
742 @ MSG_STR_UELF32        "_ELF32"
743 @ MSG_STR_UELF64        "_ELF64"
744 @ MSG_STR_USPARC        "_sparc"
745 @ MSG_STR_UX86          "_x86"
746 @ MSG_STR_TRUE          "true"

748 @ MSG_STR_CDIR_ADD      "$add"
749 @ MSG_STR_CDIR_CLEAR    "$clear"
750 @ MSG_STR_CDIR_ERROR    "$error"
751 @ MSG_STR_CDIR_MFVER    "$mapfile_version"
752 @ MSG_STR_CDIR_IF       "$if"
753 @ MSG_STR_CDIR_ELIF     "$elif"
754 @ MSG_STR_CDIR_ELSE     "$else"
755 @ MSG_STR_CDIR_ENDIF    "$endif"

757 @ MSG_STR_GROUP         "GROUP"
758 @ MSG_STR_SUNW_COMDAT   "SUNW_COMDAT"

760 @ MSG_FMT_ARMEM         "%s(%s)"
761 @ MSG_FMT_COLPATH       "%s:%s"
762 @ MSG_FMT_SYMNAM        "'%s'"
763 @ MSG_FMT_NULLSYMNAM    "%s[%d]"
764 @ MSG_FMT_STRCAT        "%s%s"

766 @ MSG_PTH_RTLD          "/usr/lib/ld.so.1"

768 @ MSG_SUNW_OST_SGS      "SUNW_OST_SGS"


771 # Section strings

773 @ MSG_SCN_BSS           ".bss"
774 @ MSG_SCN_DATA          ".data"
775 @ MSG_SCN_COMMENT       ".comment"
776 @ MSG_SCN_DEBUG         ".debug"
777 @ MSG_SCN_DEBUG_INFO    ".debug_info"
778 @ MSG_SCN_DYNAMIC       ".dynamic"
779 @ MSG_SCN_DYNSYMSORT    ".SUNW_dynsymsort"
780 @ MSG_SCN_DYNTLSSORT    ".SUNW_dyntlssort"
781 @ MSG_SCN_DYNSTR        ".dynstr"
782 @ MSG_SCN_DYNSYM        ".dynsym"
783 @ MSG_SCN_DYNSYM_SHNDX  ".dynsym_shndx"
784 @ MSG_SCN_LDYNSYM       ".SUNW_ldynsym"
785 @ MSG_SCN_LDYNSYM_SHNDX ".SUNW_ldynsym_shndx"
786 @ MSG_SCN_EX_SHARED     ".ex_shared"
787 @ MSG_SCN_EX_RANGES     ".exception_ranges"
```

```
 788 @ MSG_SCN_EXCL          ".excl"
 789 @ MSG_SCN_FINI          ".fini"
 790 @ MSG_SCN_FINIARRAY     ".fini_array"
 791 @ MSG_SCN_GOT           ".got"
 792 @ MSG_SCN_GNU_LINKONCE  ".gnu.linkonce."
 793 @ MSG_SCN_HASH          ".hash"
 794 @ MSG_SCN_INDEX         ".index"
 795 @ MSG_SCN_INIT          ".init"
 796 @ MSG_SCN_INITARRAY     ".init_array"
 797 @ MSG_SCN_INTERP        ".interp"
 798 @ MSG_SCN_LBSS          ".lbss"
 799 @ MSG_SCN_LDATA         ".ldata"
 800 @ MSG_SCN_LINE          ".line"
 801 @ MSG_SCN_LRODATA       ".lrodata"
 802 @ MSG_SCN_PLT           ".plt"
 803 @ MSG_SCN_PREINITARRAY  ".preinit_array"
 804 @ MSG_SCN_REL           ".rel"
 805 @ MSG_SCN_RELA          ".rela"
 806 @ MSG_SCN_RODATA        ".rodata"
 807 @ MSG_SCN_SBSS          ".sbss"
 808 @ MSG_SCN_SBSS2         ".sbss2"
 809 @ MSG_SCN_SDATA         ".sdata"
 810 @ MSG_SCN_SDATA2        ".sdata2"
 811 @ MSG_SCN_SHSTRTAB      ".shstrtab"
 812 @ MSG_SCN_STAB          ".stab"
 813 @ MSG_SCN_STABEXCL      ".stab.exclstr"
 814 @ MSG_SCN_STRTAB        ".strtab"
 815 @ MSG_SCN_SUNWMOVE      ".SUNW_move"
 816 @ MSG_SCN_SUNWRELOC     ".SUNW_reloc"
 817 @ MSG_SCN_SUNWSYMINFO   ".SUNW_syminfo"
 818 @ MSG_SCN_SUNWVERSION   ".SUNW_version"
 819 @ MSG_SCN_SUNWVERSYM    ".SUNW_versym"
 820 @ MSG_SCN_SUNWCAP       ".SUNW_cap"
 821 @ MSG_SCN_SUNWCAPINFO   ".SUNW_capinfo"
 822 @ MSG_SCN_SUNWCAPCHAIN  ".SUNW_capchain"
 823 @ MSG_SCN_SYMTAB        ".symtab"
 824 @ MSG_SCN_SYMTAB_SHNDX  ".symtab_shndx"
 825 @ MSG_SCN_TBSS          ".tbss"
 826 @ MSG_SCN_TDATA         ".tdata"
 827 @ MSG_SCN_TEXT          ".text"

 829 @ MSG_SYM_FINIARRAY     "finiarray"
 830 @ MSG_SYM_INITARRAY     "initarray"
 831 @ MSG_SYM_PREINITARRAY  "preinitarray"

 833 #
 834 # GNU section names
 835 #
 836 @ MSG_SCN_CTORS         ".ctors"
 837 @ MSG_SCN_DTORS         ".dtors"
 838 @ MSG_SCN_EHFRAME       ".eh_frame"
 839 @ MSG_SCN_EHFRAME_HDR   ".eh_frame_hdr"
 840 @ MSG_SCN_GCC_X_TBL     ".gcc_except_table"
 841 @ MSG_SCN_JCR           ".jcr"

 843 # Segment names for segments referenced by entrance criteria

 845 @ MSG_ENT_BSS           "bss"
 846 @ MSG_ENT_DATA          "data"
 847 @ MSG_ENT_EXTRA         "extra"
 848 @ MSG_ENT_LDATA         "ldata"
 849 @ MSG_ENT_LRODATA       "lrodata"
 850 @ MSG_ENT_NOTE          "note"
 851 @ MSG_ENT_TEXT          "text"

 853 # Symbol names
```



```
 855 @ MSG_SYM_START         "_start"
 856 @ MSG_SYM_MAIN          "main"

 858 @ MSG_SYM_FINI_U        "_fini"
 859 @ MSG_SYM_INIT_U        "_init"
 860 @ MSG_SYM_DYNAMIC       "DYNAMIC"
 861 @ MSG_SYM_DYNAMIC_U     "_DYNAMIC"
 862 @ MSG_SYM_EDATA         "edata"
 863 @ MSG_SYM_EDATA_U       "_edata"
 864 @ MSG_SYM_END           "end"
 865 @ MSG_SYM_END_U         "_end"
 866 @ MSG_SYM_ETEXT         "etext"
 867 @ MSG_SYM_ETEXT_U       "_etext"
 868 @ MSG_SYM_GOFTBL        "GLOBAL_OFFSET_TABLE_"
 869 @ MSG_SYM_GOFTBL_U      "_GLOBAL_OFFSET_TABLE_"
 870 @ MSG_SYM_PLKTBL        "PROCEDURE_LINKAGE_TABLE_"
 871 @ MSG_SYM_PLKTBL_U      "_PROCEDURE_LINKAGE_TABLE_"
 872 @ MSG_SYM_TLSGETADDR_U  "__tls_get_addr"
 873 @ MSG_SYM_TLSGETADDR_UU "___tls_get_addr"

 875 @ MSG_SYM_L_END         "END_"
 876 @ MSG_SYM_L_END_U       "_END_"
 877 @ MSG_SYM_L_START       "START_"
 878 @ MSG_SYM_L_START_U     "_START_"

 880 # Support functions

 882 @ MSG_SUP_VERSION       "ld_version"
 883 @ MSG_SUP_INPUT_DONE    "ld_input_done"

 885 @ MSG_SUP_START_64      "ld_start64"
 886 @ MSG_SUP_ATEXIT_64     "ld_atexit64"
 887 @ MSG_SUP_OPEN_64       "ld_open64"
 888 @ MSG_SUP_FILE_64       "ld_file64"
 889 @ MSG_SUP_INSEC_64      "ld_input_section64"
 890 @ MSG_SUP_SEC_64        "ld_section64"

 892 @ MSG_SUP_START         "ld_start"
 893 @ MSG_SUP_ATEXIT        "ld_atexit"
 894 @ MSG_SUP_OPEN          "ld_open"
 895 @ MSG_SUP_FILE          "ld_file"
 896 @ MSG_SUP_INSEC         "ld_input_section"
 897 @ MSG_SUP_SEC           "ld_section"

 899 #
 900 # Message previously in 'ld'
 901 #
 902 #
 903 @ _START_

 905 # System error messages

 907 @ MSG_SYS_STAT          "file %s: stat failed: %s"
 908 @ MSG_SYS_READ          "file %s: read failed: %s"
 909 @ MSG_SYS_NOTREG        "file %s: is not a regular file"

 911 # Argument processing messages

 913 @ MSG_ARG_DY_INCOMP     "%s option is incompatible with building a dynamic \
 914                          executable"
 915 @ MSG_MARG_DY_INCOMP    "%s is incompatible with building a dynamic \
 916                          executable"
 917 @ MSG_ARG_ST_INCOMP     "%s option is incompatible with building a static \
 918                          object (-dn, -r, --relocatable)"
 919 @ MSG_MARG_ST_INCOMP    "%s is incompatible with building a static \
```

```
920                          object (-dn, -r, --relocatable)"
921 @ MSG_MARG_ST_ONLYAVL    "%s is only available when building a shared object"
922 @ MSG_ARG_INCOMP         "option %s and %s are incompatible"
923 @ MSG_MARG_INCOMP        "%s and %s are incompatible"
924 @ MSG_ARG_MTONCE         "option %s appears more than once, first setting taken"
925 @ MSG_MARG_MTONCE        "%s appears more than once, first setting taken"
926 @ MSG_ARG_ILLEGAL        "option %s has illegal argument '%s'"
927 @ MSG_ARG_YP             "option -YP and -Y%c may not be specified concurrently"
928 @ MSG_ARG_STRIP          "%s specified with %s; only debugging \
929                           information stripped"
930 @ MSG_ARG_NOFILES        "no files on input command line"
931 @ MSG_ARG_NOFLTR         "option %s is only meaningful when building a filter"
932 @ MSG_ARG_NODEFLIB       "the default library search path has been suppressed, \
933                           but no runpaths have been specified via %s"
934 @ MSG_ARG_NOENTRY        "entry point symbol '%s' is undefined"
935 @ MSG_ARG_UNSUPPORTED    "option %s is no longer supported; ignored"
936 @ MSG_MARG_ONLY          "option %s can only be used with a %s"
937 @ MSG_ARG_UNKNOWN        "unrecognized option '-%c'"
938 @ MSG_ARG_LONG_UNKNOWN   "unrecognized option '%s'"
939 @ MSG_ARG_USEHELP        "use the -z help option for usage information"


942 @ MSG_ARG_FLAGS          "flags processing errors"
943 @ MSG_ARG_FILES          "file processing errors. No output written to %s"
944 @ MSG_ARG_SYM_WARN       "symbol referencing errors"
945 @ MSG_ARG_SYM_FATAL      "symbol referencing errors. No output written to %s"
946 @ MSG_ARG_AR_GRP_OLAP    "%s cannot be nested"
947 @ MSG_ARG_AR_GRP_BAD     "%s used without corresponding %s"


950 # Messages used to refer to options where there is more than
951 # one name accepted.

953 @ MSG_MARG_AR_GRPS       "archive rescan groups \
954                           (-z rescan-start, -(, --start-group)"
955 @ MSG_MARG_AR_GRP_END    "archive rescan group end option \
956                           (-z rescan-end, -), --end-group)"
957 @ MSG_MARG_AR_GRP_START "archive rescan group start option \
958                           (-z rescan-start, -(, --start-group)"
959 @ MSG_MARG_ENTRY         "entry point option (-e, --entry)"
960 @ MSG_MARG_FILTER_AUX    "auxiliary filter option (-f, --auxiliary)"
961 @ MSG_MARG_FILTER        "filter option (-F, --filter)"
962 @ MSG_MARG_OUTFILE       "output object option (-o, --output)"
963 @ MSG_MARG_REL           "relocatable object option (-r, --relocatable)"
964 @ MSG_MARG_RPATH         "runpath option (-R, -rpath)"
965 @ MSG_MARG_SO            "shared object option (-G, -shared)"
966 @ MSG_MARG_SONAME        "soname option (-h, --soname)"
967 @ MSG_MARG_STRIP         "strip option (-s, --strip-all)"

969 # Entrance criteria messages

971 @ MSG_ENT_MAP_FMT_TIL_1 "\t\t%s\n\n"
972 @ MSG_ENT_MAP_TITLE_1   "LINK EDITOR MEMORY MAP"

974 #
975 # TRANSLATION_NOTE -- Entry map header
976 #
977 # The next message is a format string for a title. The title is composed of
978 # two lines. In C locale, it would look like:
979 #
980 #       output          input           new
981 #       section         section         displacement    size
982 #
983 # The \t characters are used for alignment.  (output section), (input section),
984 # and (new displacement) have to be aligned.
985 #
```



```
986 @ MSG_ENT_MAP_FMT_TIL_2 "\n%s\t\t%s\t\t%s\n%s\t\t%s\t\t%s\t%s\n\n"
987 @ MSG_ENT_MAP_FMT_TIL_3 "\n%s\t\t%s\t\t%s\n%s\t\t%s\t\t%s\t\t%s\n\n"
988 @ MSG_ENT_ITM_OUTPUT     "output"
989 @ MSG_ENT_ITM_INPUT      "input"
990 @ MSG_ENT_ITM_NEW        "new"
991 @ MSG_ENT_ITM_SECTION    "section"
992 @ MSG_ENT_ITM_DISPMNT    "displacement"
993 @ MSG_ENT_ITM_SIZE       "size"
994 @ MSG_ENT_ITM_VIRTUAL    "virtual"
995 @ MSG_ENT_ITM_ADDRESS    "address"

997 @ MSG_ENT_MAP_ENTRY_1    "%-8.8s\t\t\t%08.2llx\t%08.2llx\n"
998 @ MSG_ENT_MAP_ENTRY_2    "\t\t%-8.8s\t%08.2llx\t%08.2llx %s\n"

1000 #
1001 # TRANSLATION_NOTE -- multiple defined symbol table header
1002 #
1003 # In C locale, an example output is:
1004 #
1005 #               MULTIPLY DEFINED SYMBOLS
1006 #
1007 #
1008 #symbol                          definition used     also defined in
1009 #
1010 #variable1                          main.o
1011 #                                                      ./libfred.so
1012 @ MSG_ENT_MUL_FMT_TIL_0 "\n\n\t\t%s\n\n\n"
1013 @ MSG_ENT_MUL_TIL_0     "MULTIPLY DEFINED SYMBOLS"

1015 #
1016 # TRANSLATION_NOTE -- This is the format string for:
1017 #
1018 #symbol                          definition used     also defined in
1019 #
1020 @ MSG_ENT_MUL_FMT_TIL_1 "%s\t\t\t\t  %s     %s\n\n"
1021 @ MSG_ENT_MUL_ITM_SYM   "symbol"
1022 @ MSG_ENT_MUL_ITM_DEF_0 "definition used"
1023 @ MSG_ENT_MUL_ITM_DEF_1 "also defined in"

1025 #
1026 # TRANSLATION_NOTE -- This is the format string for the second item:
1027 #
1028 @ MSG_ENT_MUL_ENTRY_1   "%-35s %s\n"

1030 #
1031 # TRANSLATION_NOTE -- This is the format string for the third item:
1032 #
1033 @ MSG_ENT_MUL_ENTRY_2   "\t\t\t\t\t\t\t%s\n"

1035 @ MSG_ENT_NOSEC_1       "mapfile: %s segment: section '%s' does not appear \
1036                           in mapfile specified input file(s)"
1037 @ MSG_ENT_NOSEC_2       "mapfile: %s segment: section '%s' does not appear \
1038                           in any input file"

1040 # Library messages

1042 @ MSG_LIB_NOTFOUND      "library -l%s: not found"
1043 @ MSG_LIB_MALFORM       "LD_LIBRARY_PATH malformed"
1044 @ MSG_LIB_BADYP         "-YP library path malformed"


1047 # Mapfile processing messages

1049 @ MSG_MAP_BADAUTORED    "%s: %llu: auto-reduction ('*') can only be used in \
1050                           hidden/local, or eliminate scope"
1051 @ MSG_MAP_BADFLAG       "%s: %llu: badly formed section flags '%s'"
```

```
1052 @ MSG_MAP_BADBNAME      "%s: %llu: basename cannot contain path \
1053                          separator (’/’): %s"
1054 @ MSG_MAP_BADONAME      "%s: %llu: object name cannot contain path \
1055                          separator (’/’): %s"
1056 @ MSG_MAP_REDEFATT      "%s: %llu: redefining %s attribute for ’%s’"
1057 @ MSG_MAP_PREMEOF       "%s: %llu: premature EOF"
1058 @ MSG_MAP_ILLCHAR       "%s: %llu: illegal character ’\\%03o’"
1059 @ MSG_MAP_MALFORM       "%s: %llu: malformed entry"
1060 @ MSG_MAP_NONLOAD       "%s: %llu: %s not allowed on non-LOAD segments"
1061 @ MSG_MAP_NOSTACK1      "%s: %llu: %s not allowed on STACK segment"
1062 @ MSG_MAP_MOREONCE      "%s: %llu: %s set more than once on same line"
1063 @ MSG_MAP_NOTERM        "%s: %llu: unterminated quoted string: %s"
1064 @ MSG_MAP_SECINSEG      "%s: %llu: section within segment ordering done on \
1065                          a non-existent segment ’%s’"
1066 @ MSG_MAP_UNEXINHERIT   "%s: %llu: unnamed version cannot inherit from other \
1067                          versions: %s"
1068 @ MSG_MAP_UNEXTOK       "%s: %llu: unexpected occurrence of ’%c’ token"

1070 @ MSG_MAP_SEGEMPLOAD    "%s: %llu: empty segment must be of type LOAD or NULL"
1071 @ MSG_MAP_SEGEMPEXE     "%s: %llu: a LOAD empty segment definition is only \
1072                          allowed when creating a dynamic executable"
1073 @ MSG_MAP_SEGEMPATT     "%s: %llu: a LOAD empty segment must have an address \
1074                          and size"
1075 @ MSG_MAP_SEGEMPNOATT   "%s: %llu: a NULL empty segment must not have an \
1076                          address or size"
1077 @ MSG_MAP_SEGEMPSEC     "%s: %llu: empty segment can not have sections \
1078                          assigned to it"
1079 @ MSG_MAP_SEGEMNOPERM   "%s: %llu: empty segment must not have \
1080                          p_flags set: 0x%x"

1082 @ MSG_MAP_CNTADDRORDER  "%s: %llu: segment cannot have an explicit address \
1083                          and also be in the SEGMENT_ORDER list: %s"
1084 @ MSG_MAP_CNTDISSEG     "%s: %llu: segment cannot be disabled: %s"
1085 @ MSG_MAP_DUPNAMENT     "%s: %llu: cannot redefine entrance criteria: %s"
1086 @ MSG_MAP_DUPORDSEG     "%s: %llu: segment is already in %s list: %s"
1087 @ MSG_MAP_DUP_OS_ORD    "%s: %llu: section is already in OS_ORDER list: %s"
1088 @ MSG_MAP_DUP_IS_ORD    "%s: %llu: entrance criteria is already in \
1089                          IS_ORDER list: %s"
1090 @ MSG_MAP_UNKENT        "%s: %llu: unknown entrance criteria \
1091                          (ASSIGN_SECTION): %s"
1092 @ MSG_MAP_UNKSEG        "%s: %llu: unknown segment: %s"
1093 @ MSG_MAP_UNKSYMDEF     "%s: %llu: unknown symbol definition: %s"
1094 @ MSG_MAP_UNKSEGTYP     "%s: %llu: unknown internal segment type %d"
1095 @ MSG_MAP_UNKSOTYP      "%s: %llu: unknown shared object type: %s"
1096 @ MSG_MAP_UNKSEGATT     "%s: %llu: unknown segment attribute: %s"
1097 @ MSG_MAP_UNKSEGFLG     "%s: %llu: unknown segment flag: ?%c"
1098 @ MSG_MAP_UNKSECTYP     "%s: %llu: unknown section type: %s"

1100 @ MSG_MAP_SEGSIZE       "%s: %lld: existing segment size symbols cannot \
1101                          be reset: %s"
1102 @ MSG_MAP_SEGADDR       "%s: %llu: segment address or length ’%s’ %s"
1103 @ MSG_MAP_BADCAPVAL     "%s: %llu: bad capability value: %s"
1104 @ MSG_MAP_UNKCAPATTR    "%s: %llu: unknown capability attribute ’%s’"
1105 @ MSG_MAP_EMPTYCAP      "%s: %llu: empty capability definition; ignored"

1107 @ MSG_MAP_SYMDEF1       "%s: %llu: symbol ’%s’ is already defined in file: \
1108                          %s: %s"
1109 @ MSG_MAP_SYMDEF2       "%s: %llu: symbol ’%s’: %s"

1111 @ MSG_MAP_EXPSCOL       "%s: %llu: expected a ’;’"
1112 @ MSG_MAP_EXPEQU        "%s: %llu: expected a ’=’, ’:’, ’|’, or ’@’"
1113 @ MSG_MAP_EXPSEGATT     "%s: %llu: expected one or more segment attributes \
1114                          after an ’=’"
1115 @ MSG_MAP_EXPSEGNAM     "%s: %llu: expected a segment name at the beginning \
1116                          of a line"
1117 @ MSG_MAP_EXPSEGTYPE    "%s: %llu: %s segment cannot be used with %s \
```



```
1118                          directive: %s"
1119 @ MSG_MAP_EXPSYM_1      "%s: %llu: expected a symbol name after ’@’"
1120 @ MSG_MAP_EXPSYM_2      "%s: %llu: expected a symbol name after ’{’"
1121 @ MSG_MAP_EXPSEC        "%s: %llu: expected a section name after ’|’"
1122 @ MSG_MAP_EXPSO         "%s: %llu: expected a shared object definition \
1123                          after ’-’"
1124 @ MSG_MAP_MULTFILTEE    "%s: %llu: multiple filtee definitions are unsupported"
1125 @ MSG_MAP_NOFILTER      "%s: %llu: filtee definition required"
1126 @ MSG_MAP_BADSF1        "%s: %llu: unknown software capabilities: 0x%llx; \
1127                          ignored"
1128 @ MSG_MAP_INADDR32SF1   "%s: %llu: software capability ADDR32: is ineffective \
1129                          when building 32-bit object: ignored"
1130 @ MSG_MAP_NOINTPOSE     "%s: %llu: interposition symbols can only be defined \
1131                          when building a dynamic executable"
1132 @ MSG_MAP_NOEXVLSZ      "%s: %llu: value and size attributes are incompatible \
1133                          with extern or parent symbols"
1134 @ MSG_MAP_FLTR_ONLYAVL  "%s: %llu: symbol filtering is only available when \
1135                          building a shared object"

1137 @ MSG_MAP_SEGSAME       "segments ’%s’ and ’%s’ have the same assigned \
1138                          virtual address"
1139 @ MSG_MAP_EXCLIMIT      "exceeds internal limit"
1140 @ MSG_MAP_NOBADFRM      "number is badly formed"

1142 @ MSG_MAP_SEGTYP        "segment type"
1143 @ MSG_MAP_SEGVADDR      "segment virtual address"
1144 @ MSG_MAP_SEGPHYS       "segment physical address"
1145 @ MSG_MAP_SEGLEN        "segment length"
1146 @ MSG_MAP_SEGFLAG       "segment flags"
1147 @ MSG_MAP_SEGALIGN      "segment alignment"
1148 @ MSG_MAP_SEGROUND      "segment rounding"

1150 @ MSG_MAP_SECTYP        "section type"
1151 @ MSG_MAP_SECFLAG       "section flags"
1152 @ MSG_MAP_SECNAME       "section name"

1154 @ MSG_MAP_SYMVAL        "symbol value"
1155 @ MSG_MAP_SYMSIZE       "symbol size"

1157 @ MSG_MAP_DIFF_SYMVAL   "symbol values differ"
1158 @ MSG_MAP_DIFF_SYMSZ    "symbol sizes differ"
1159 @ MSG_MAP_DIFF_SYMTYP   "symbol types differ"
1160 @ MSG_MAP_DIFF_SYMNDX   "symbol indexes differ"
1161 @ MSG_MAP_DIFF_SYMLCL   "symbol scope conflict against local and non-local"
1162 @ MSG_MAP_DIFF_SYMGLOB  "symbol scope conflict against singleton/exported"
1163 @ MSG_MAP_DIFF_SYMPROT  "symbol scope conflict against protected"
1164 @ MSG_MAP_DIFF_SYMVER   "symbol version conflict"
1165 @ MSG_MAP_DIFF_SYMMUL   "symbol multiple definition"
1166 @ MSG_MAP_DIFF_SNGLDIR  "singleton scope and direct declaration are \
1167                          incompatible"
1168 @ MSG_MAP_DIFF_PROTNDIR "protected scope and no-direct declaration \
1169                          are incompatible"


1172 @ MSG_MAP_SECORDER      "section ordering requested, but no matching section \
1173                          found: segment: %s section: %s"


1176 # Mapfile Directives

1178 @ MSG_MAP_EXP_ATTR      "%s: %llu: expected attribute name (%s), or \
1179                          terminator (’;’, ’}’): %s"
1180 @ MSG_MAP_EXP_CAPMASK   "%s: %llu: expected capability name, integer value, or \
1181                          terminator (’;’, ’}’): %s"
1182 @ MSG_MAP_EXP_CAPNAME   "%s: %llu: expected name, or terminator (’;’, ’}’): %s"
1183 @ MSG_MAP_EXP_CAPID     "%s: %llu: expected name, or ’{’ following %s: %s"
```

```
1184 @ MSG_MAP_EXP_CAPHW     "%s: %llu: expected hardware capability, or \
1185                          terminator (';', '}'): %s"
1186 @ MSG_MAP_EXP_CAPSF     "%s: %llu: expected software capability, or \
1187                          terminator (';', '}'): %s"
1188 @ MSG_MAP_EXP_EQ        "%s: %llu: expected '=' following %s: %s"
1189 @ MSG_MAP_EXP_EQ_ALL    "%s: %llu: expected '=', '+=', or '-=' following %s: %s"
1190 @ MSG_MAP_EXP_EQ_PEQ    "%s: %llu: expected '=' following %s: %s"
1191 @ MSG_MAP_EXP_DIR       "%s: %llu: expected mapfile directive (%s): %s"
1192 @ MSG_MAP_SFLG_EXBANG   "%s: %llu: '!' appears without corresponding flag"
1193 @ MSG_MAP_EXP_FILNAM    "%s: %llu: expected file name following %s: %s"
1194 @ MSG_MAP_EXP_FILPATH   "%s: %llu: expected file path following %s: %s"
1195 @ MSG_MAP_EXP_INT       "%s: %llu: expected integer value following %s: %s"
1196 @ MSG_MAP_EXP_LBKT      "%s: %llu: expected '{' following %s: %s"
1197 @ MSG_MAP_EXP_OBJNAM    "%s: %llu: expected object name following %s: %s"
1198 @ MSG_MAP_SFLG_ONEBANG  "%s: %llu: '!' can only be specified once per flag"
1199 @ MSG_MAP_EXP_SECFLAG   "%s: %llu: expected section flag (%s), '!', or \
1200                          terminator (';', '}'): %s"
1201 @ MSG_MAP_EXP_SECNAM    "%s: %llu: expected section name following %s: %s"
1202 @ MSG_MAP_EXP_SEGFLAG   "%s: %llu: expected segment flag (%s), or \
1203                          terminator (';', '}'): %s"
1204 @ MSG_MAP_EXP_ECNAM     "%s: %llu: expected entrance criteria (ASSIGN_SECTION) \
1205                          name, or terminator (';', '}'): %s"
1206 @ MSG_MAP_EXP_SEGNAM    "%s: %llu: expected segment name following %s: %s"
1207 @ MSG_MAP_EXP_SEM       "%s: %llu: expected ';' to terminate %s: %s"
1208 @ MSG_MAP_EXP_SEMLBKT   "%s: %llu: expected ';' or '{' following %s: %s"
1209 @ MSG_MAP_EXP_SEMRBKT   "%s: %llu: expected ';' or '}' to terminate %s: %s"
1210 @ MSG_MAP_EXP_SHTYPE    "%s: %llu: expected section type: %s"
1211 @ MSG_MAP_EXP_SYM       "%s: %llu: expected symbol name, symbol scope, \
1212                          or '*': %s"
1213 @ MSG_MAP_EXP_SYMEND    "%s: %llu: expected inherited version name, or \
1214                          terminator (';'): %s"
1215 @ MSG_MAP_EXP_SYMDELIM  "%s: %llu: expected one of ':', ';', or '{': %s"
1216 @ MSG_MAP_EXP_SYMFLAG   "%s: %llu: expected symbol flag (%s), or \
1217                          terminator (';', '}'): %s"
1218 @ MSG_MAP_EXP_SYMNAM    "%s: %llu: expected symbol name following %s: %s"
1219 @ MSG_MAP_EXP_SYMSCOPE  "%s: %llu: expected symbol scope (%s): %s"
1220 @ MSG_MAP_EXP_SYMTYPE   "%s: %llu: expected symbol type (%s): %s"
1221 @ MSG_MAP_EXP_VERSION   "%s: %llu: expected version name following %s: %s"
1222 @ MSG_MAP_BADEXTRA      "%s: %llu: unexpected text found following %s directive"
1223 @ MSG_MAP_VALUELIMIT    "%s: %llu: numeric value exceeds word size: %s"
1224 @ MSG_MAP_MALVALUE      "%s: %llu: malformed numeric value: %s"
1225 @ MSG_MAP_BADVALUETAIL  "%s: %llu: unexpected characters following numeric \
1226                          constant: %s"
1227 @ MSG_MAP_WSNEEDED      "%s: %llu: whitespace needed before token: %s"
1228 @ MSG_MAP_BADCHAR       "%s: %llu: unexpected text: %s"
1229 @ MSG_MAP_BADKWQUOTE    "%s: %llu: mapfile keywords should not be quoted: %s"
1230 @ MSG_MAP_CDIR_NOTBOL   "%s: %llu: mapfile control directive not at start of \
1231                          line: %s"
1232 @ MSG_MAP_NOATTR        "%s: %llu: %s specified no attributes (empty {})"
1233 @ MSG_MAP_NOVALUES      "%s: %llu: %s specified without values"
1234 @ MSG_MAP_INTERR        "<internal error>"
1235 @ MSG_MAP_ISORDVER      "%s: %llu: version 0 mapfile ?O flag and version 1 \
1236                          segment IS_ORDER attribute are mutually exclusive: %s"
1237 @ MSG_MAP_SYMATTR       "symbol attributes";

1239 # Mapfile Control Directives

1241 @ MSG_MAP_CDIR_BADVDIR  "%s: %llu: $mapfile_version directive must specify \
1242                          version 2 or higher: %d"
1243 @ MSG_MAP_CDIR_BADVER   "%s: %llu: unknown mapfile version: %d"
1244 @ MSG_MAP_CDIR_REPVER   "%s: %llu: $mapfile_version must be first directive \
1245                          in file"
1246 @ MSG_MAP_CDIR_REQARG   "%s: %llu: %s directive requires an argument"
1247 @ MSG_MAP_CDIR_REQNOARG "%s: %llu: %s directive does not accept arguments"
1248 @ MSG_MAP_CDIR_BAD      "%s: %llu: unrecognized mapfile control directive"
1249 @ MSG_MAP_CDIR_NOIF     "%s: %llu: %s directive used without opening $if"
```



```
1250 @ MSG_MAP_CDIR_ELSE     "%s: %llu: %s directive preceded by $else on line %d"
1251 @ MSG_MAP_CDIR_NOEND    "%s: %llu: EOF encountered without closing $endif \
1252                          for $if on line %d"
1253 @ MSG_MAP_CDIR_ERROR    "%s: %llu: error: %s"


1256 # Mapfile Conditional Expressions

1258 @ MSG_MAP_CEXP_TOKERR   "%s: %llu: syntax error in conditional expression at: %s
1259 @ MSG_MAP_CEXP_SEMERR   "%s: %llu: malformed conditional expression"
1260 @ MSG_MAP_CEXP_BADOPUSE "%s: %llu: invalid operator use in conditional \
1261                          expression"
1262 @ MSG_MAP_CEXP_UNBALPAR "%s: %llu: unbalanced parenthesis in conditional \
1263                          expression"
1264 @ MSG_MAP_BADCESC       "%s: %llu: unrecognized escape in double quoted \
1265                          token: \\%c\n"

1267 # Generic error diagnostic labels

1269 @ MSG_STR_NULL          "(null)"

1271 @ MSG_DBG_DFLT_FMT      "debug: "
1272 @ MSG_DBG_AOUT_FMT      "debug: a.out: "
1273 @ MSG_DBG_NAME_FMT      "debug: %s: "

1275 # -z assert-deflib strings

1277 @ MSG_ARG_ASSDEFLIB_MALFORMED   "library name malformed: %s"
1278 @ MSG_ARG_ASSDEFLIB_FOUND       "dynamic library found on default search path \
1279                                 (%s): lib%s.so"

1281 @ _END_


1284 # Software identification.  Note, the SGU strings is historic, and has
1285 # little relevance.  It is preserved as applications have used this
1286 # string to identify the Solaris link-editor.

1288 @ MSG_SGS_ID            "ld: Software Generation Utilities - \
1289                          Solaris Link Editors: "

1291 # The following strings represent reserved words, files, pathnames and symbols.
1292 # Reference to this strings is via the MSG_ORIG() macro, and thus no message
1293 # translation is required.

1295 @ MSG_DBG_FOPEN_MODE    "w"

1297 @ MSG_DBG_CLS32_FMT     "32: "
1298 @ MSG_DBG_CLS64_FMT     "64: "

1300 @ MSG_STR_PATHTOK       ";:"
1301 @ MSG_STR_AOUT          "a.out"

1303 @ MSG_STR_LIB_A         "%s/lib%s.a"
1304 @ MSG_STR_LIB_SO        "%s/lib%s.so"
1305 @ MSG_STR_PATH          "%s/%s"
1306 @ MSG_STR_STRNL         "%s\n"
1307 @ MSG_STR_NL            "\n"
1308 @ MSG_STR_CAPGROUPID    "CAP_GROUP_%d"

1310 @ MSG_STR_LD_DYNAMIC    "dynamic"
1311 @ MSG_STR_SYMBOLIC      "symbolic"
1312 @ MSG_STR_ELIMINATE     "eliminate"
1313 @ MSG_STR_LOCAL         "local"
1314 @ MSG_STR_PROGBITS      "progbits"
1315 @ MSG_STR_SYMTAB        "symtab"
```

```
1316 @ MSG_STR_DYNSYM        "dynsym"
1317 @ MSG_STR_REL           "rel"
1318 @ MSG_STR_RELA          "rela"
1319 @ MSG_STR_STRTAB        "strtab"
1320 @ MSG_STR_HASH          "hash"
1321 @ MSG_STR_LIB           "lib"
1322 @ MSG_STR_NOTE          "note"
1323 @ MSG_STR_NOBITS        "nobits"
1324 @ MSG_STR_HWCAP_1       "hwcap_1"
1325 @ MSG_STR_SFCAP_1       "sfcap_1"
1326 @ MSG_STR_SOEXT         ".so"

1328 @ MSG_STR_OPTIONS       "3:6:abc:d:e:f:h:il:mo:p:rstu:z:B:CD:F:GI:L:M:N:P:Q:R:\
1329                          S:VW:Y:?"

1331 # Argument processing strings

1333 @ MSG_ARG_3             "-3"
1334 @ MSG_ARG_6             "-6"
1335 @ MSG_ARG_A             "-a"
1336 @ MSG_ARG_B             "-b"
1337 @ MSG_ARG_CB            "-B"
1338 @ MSG_ARG_BDIRECT       "-Bdirect"
1339 @ MSG_ARG_BDYNAMIC      "-Bdynamic"
1340 @ MSG_ARG_BELIMINATE    "-Beliminate"
1341 @ MSG_ARG_BGROUP        "-Bgroup"
1342 @ MSG_ARG_BLOCAL        "-Blocal"
1343 @ MSG_ARG_BNODIRECT     "-Bnodirect"
1344 @ MSG_ARG_BSYMBOLIC     "-Bsymbolic"
1345 @ MSG_ARG_BTRANSLATOR   "-Btranslator"
1346 @ MSG_ARG_C             "-c"
1347 @ MSG_ARG_D             "-d"
1348 @ MSG_ARG_DY            "-dy"
1349 @ MSG_ARG_CI            "-I"
1350 @ MSG_ARG_CN            "-N"
1351 @ MSG_ARG_P             "-p"
1352 @ MSG_ARG_CP            "-P"
1353 @ MSG_ARG_CQ            "-Q"
1354 @ MSG_ARG_CY            "-Y"
1355 @ MSG_ARG_CYL           "-YL"
1356 @ MSG_ARG_CYP           "-YP"
1357 @ MSG_ARG_CYU           "-YU"
1358 @ MSG_ARG_Z             "-z"
1359 @ MSG_ARG_ZDEFNODEF     "-z[defs|nodefs]"
1360 @ MSG_ARG_ZASLR         "-zaslr"
1361 @ MSG_ARG_ZGUIDE        "-zguidance"
1362 @ MSG_ARG_ZNODEF        "-znodefs"
1363 @ MSG_ARG_ZNOINTERP     "-znointerp"
1364 @ MSG_ARG_ZRELAXRELOC   "-zrelaxreloc"
1365 @ MSG_ARG_ZNORELAXRELOC "-znorelaxreloc"
1366 @ MSG_ARG_ZTEXT         "-ztext"
1367 @ MSG_ARG_ZTEXTOFF      "-ztextoff"
1368 @ MSG_ARG_ZTEXTWARN     "-ztextwarn"
1369 @ MSG_ARG_ZTEXTALL      "-z[text|textwarn|textoff]"
1370 @ MSG_ARG_ZLOADFLTR     "-zloadfltr"
1371 @ MSG_ARG_ZCOMBRELOC    "-zcombreloc"
1372 @ MSG_ARG_ZSYMBOLCAP    "-zsymbolcap"
1373 @ MSG_ARG_ZFATWNOFATW   "-z[fatal-warnings|nofatalwarnings]"

1375 @ MSG_ARG_ABSEXEC       "absexec"
1376 @ MSG_ARG_ALTEXEC64     "altexec64"
1377 @ MSG_ARG_ASLR          "aslr"
1378 @ MSG_ARG_NOCOMPSTRTAB  "nocompstrtab"
1379 @ MSG_ARG_GROUPPERM     "groupperm"
1380 @ MSG_ARG_NOGROUPPERM   "nogroupperm"
1381 @ MSG_ARG_LAZYLOAD      "lazyload"
```



```
1382 @ MSG_ARG_NOLAZYLOAD    "nolazyload"
1383 @ MSG_ARG_INTERPOSE     "interpose"
1384 @ MSG_ARG_DIRECT        "direct"
1385 @ MSG_ARG_NODIRECT      "nodirect"
1386 @ MSG_ARG_IGNORE        "ignore"
1387 @ MSG_ARG_RECORD        "record"
1388 @ MSG_ARG_INITFIRST     "initfirst"
1389 @ MSG_ARG_INITARRAY     "initarray="
1390 @ MSG_ARG_FINIARRAY     "finiarray="
1391 @ MSG_ARG_PREINITARRAY  "preinitarray="
1392 @ MSG_ARG_RTLDINFO      "rtldinfo="
1393 @ MSG_ARG_DTRACE        "dtrace="
1394 @ MSG_ARG_TRANSLATOR    "translator"
1395 @ MSG_ARG_NOOPEN        "nodlopen"
1396 @ MSG_ARG_NOW           "now"
1397 @ MSG_ARG_ORIGIN        "origin"
1398 @ MSG_ARG_DEFS          "defs"
1399 @ MSG_ARG_NODEFS        "nodefs"
1400 @ MSG_ARG_NODUMP        "nodump"
1401 @ MSG_ARG_NOVERSION     "noversion"
1402 @ MSG_ARG_TEXT          "text"
1403 @ MSG_ARG_TEXTOFF       "textoff"
1404 @ MSG_ARG_TEXTWARN      "textwarn"
1405 @ MSG_ARG_MULDEFS       "muldefs"
1406 @ MSG_ARG_NODELETE      "nodelete"
1407 @ MSG_ARG_NOINTERP      "nointerp"
1408 @ MSG_ARG_NOPARTIAL     "nopartial"
1409 @ MSG_ARG_NORELOC       "noreloc"
1410 @ MSG_ARG_REDLOCSYM     "redlocsym"
1411 @ MSG_ARG_VERBOSE       "verbose"
1412 @ MSG_ARG_WEAKEXT       "weakextract"
1413 @ MSG_ARG_LOADFLTR      "loadfltr"
1414 @ MSG_ARG_ALLEXTRT      "allextract"
1415 @ MSG_ARG_DFLEXTRT      "defaultextract"
1416 @ MSG_ARG_COMBRELOC     "combreloc"
1417 @ MSG_ARG_NOCOMBRELOC   "nocombreloc"
1418 @ MSG_ARG_NODEFAULTLIB  "nodefaultlib"
1419 @ MSG_ARG_ENDFILTEE     "endfiltee"
1420 @ MSG_ARG_LD32          "ld32="
1421 @ MSG_ARG_LD64          "ld64="
1422 @ MSG_ARG_RESCAN        "rescan"
1423 @ MSG_ARG_RESCAN_NOW    "rescan-now"
1424 @ MSG_ARG_RESCAN_START  "rescan-start"
1425 @ MSG_ARG_RESCAN_END    "rescan-end"
1426 @ MSG_ARG_GUIDE         "guidance"
1427 @ MSG_ARG_NOLDYNSYM     "noldynsym"
1428 @ MSG_ARG_RELAXRELOC    "relaxreloc"
1429 @ MSG_ARG_NORELAXRELOC  "norelaxreloc"
1430 @ MSG_ARG_NOSIGHANDLER  "nosighandler"
1431 @ MSG_ARG_GLOBAUDIT     "globalaudit"
1432 @ MSG_ARG_TARGET        "target="
1433 @ MSG_ARG_WRAP          "wrap="
1434 @ MSG_ARG_FATWARN       "fatal-warnings"
1435 @ MSG_ARG_NOFATWARN     "nofatal-warnings"
1436 @ MSG_ARG_HELP          "help"
1437 @ MSG_ARG_GROUP         "group"
1438 @ MSG_ARG_REDUCE        "reduce"
1439 @ MSG_ARG_STATIC        "static"
1440 @ MSG_ARG_SYMBOLCAP     "symbolcap"
1441 @ MSG_ARG_DEFERRED      "deferred"
1442 @ MSG_ARG_NODEFERRED    "nodeferred"
1443 @ MSG_ARG_ASSDEFLIB     "assert-deflib"

1445 @ MSG_ARG_LCOM          "L,"
1446 @ MSG_ARG_PCOM          "P,"
1447 @ MSG_ARG_UCOM          "U,"
```

```
1449 @ MSG_ARG_T_RPATH       "rpath"
1450 @ MSG_ARG_T_SHARED      "shared"
1451 @ MSG_ARG_T_SONAME      "soname"
1452 @ MSG_ARG_T_WL          "l,-"

1454 @ MSG_ARG_T_AUXFLTR     "-auxiliary"
1455 @ MSG_ARG_T_MULDEFS     "-allow-multiple-definition"
1456 @ MSG_ARG_T_INTERP      "-dynamic-linker"
1457 @ MSG_ARG_T_ENDGROUP    "-end-group"
1458 @ MSG_ARG_T_ENTRY       "-entry"
1459 @ MSG_ARG_T_STDFLTR     "-filter"
1460 @ MSG_ARG_T_FATWARN     "-fatal-warnings"
1461 @ MSG_ARG_T_NOFATWARN   "-no-fatal-warnings"
1462 @ MSG_ARG_T_HELP        "-help"
1463 @ MSG_ARG_T_LIBRARY     "-library"
1464 @ MSG_ARG_T_LIBPATH     "-library-path"
1465 @ MSG_ARG_T_NOUNDEF     "-no-undefined"
1466 @ MSG_ARG_T_NOWHOLEARC  "-no-whole-archive"
1467 @ MSG_ARG_T_OUTPUT      "-output"
1468 @ MSG_ARG_T_RELOCATABLE "-relocatable"
1469 @ MSG_ARG_T_STARTGROUP  "-start-group"
1470 @ MSG_ARG_T_STRIP       "-strip-all"
1471 @ MSG_ARG_T_UNDEF       "-undefined"
1472 @ MSG_ARG_T_VERSION     "-version"
1473 @ MSG_ARG_T_WHOLEARC    "-whole-archive"
1474 @ MSG_ARG_T_WRAP        "-wrap"
1475 @ MSG_ARG_T_OPAR        "("
1476 @ MSG_ARG_T_CPAR        ")"

1478 @ MSG_ARG_ENABLED       "enabled"
1479 @ MSG_ARG_DISABLED      "disabled"
1480 @ MSG_ARG_ENABLE        "enable"
1481 @ MSG_ARG_DISABLE       "disable"
1482 #endif /* ! codereview */

1484 # -z guidance=item strings
1485 @ MSG_ARG_GUIDE_DELIM          ",: \t"
1486 @ MSG_ARG_GUIDE_NO_ALL         "noall"
1487 @ MSG_ARG_GUIDE_NO_DEFS        "nodefs"
1488 @ MSG_ARG_GUIDE_NO_DIRECT      "nodirect"
1489 @ MSG_ARG_GUIDE_NO_LAZYLOAD    "nolazyload"
1490 @ MSG_ARG_GUIDE_NO_MAPFILE     "nomapfile"
1491 @ MSG_ARG_GUIDE_NO_TEXT        "notext"
1492 @ MSG_ARG_GUIDE_NO_UNUSED      "nounused"

1494 # Environment variable strings

1496 @ MSG_LD_RUN_PATH       "LD_RUN_PATH"
1497 @ MSG_LD_LIBPATH_32     "LD_LIBRARY_PATH_32"
1498 @ MSG_LD_LIBPATH_64     "LD_LIBRARY_PATH_64"
1499 @ MSG_LD_LIBPATH        "LD_LIBRARY_PATH"

1501 @ MSG_LD_NOVERSION_32   "LD_NOVERSION_32"
1502 @ MSG_LD_NOVERSION_64   "LD_NOVERSION_64"
1503 @ MSG_LD_NOVERSION      "LD_NOVERSION"

1505 @ MSG_SGS_SUPPORT_32    "SGS_SUPPORT_32"
1506 @ MSG_SGS_SUPPORT_64    "SGS_SUPPORT_64"
1507 @ MSG_SGS_SUPPORT       "SGS_SUPPORT"


1510 # Symbol names

1512 @ MSG_SYM_LIBVER_U      "_lib_version"
```



```
1515 # Mapfile tokens

1517 @ MSG_MAP_LOAD          "load"
1518 @ MSG_MAP_NOTE          "note"
1519 @ MSG_MAP_NULL          "null"
1520 @ MSG_MAP_STACK         "stack"
1521 @ MSG_MAP_ADDVERS       "addvers"
1522 @ MSG_MAP_FUNCTION      "function"
1523 @ MSG_MAP_DATA          "data"
1524 @ MSG_MAP_COMMON        "common"
1525 @ MSG_MAP_PARENT        "parent"
1526 @ MSG_MAP_EXTERN        "extern"
1527 @ MSG_MAP_DIRECT        "direct"
1528 @ MSG_MAP_NODIRECT      "nodirect"
1529 @ MSG_MAP_FILTER        "filter"
1530 @ MSG_MAP_AUXILIARY     "auxiliary"
1531 @ MSG_MAP_OVERRIDE      "override"
1532 @ MSG_MAP_INTERPOSE     "interpose"
1533 @ MSG_MAP_DYNSORT       "dynsort"
1534 @ MSG_MAP_NODYNSORT     "nodynsort"

1536 @ MSG_MAPKW_ALIGN               "ALIGN"
1537 @ MSG_MAPKW_ALLOC               "ALLOC"
1538 @ MSG_MAPKW_ALLOW               "ALLOW"
1539 @ MSG_MAPKW_AMD64_LARGE         "AMD64_LARGE"
1540 @ MSG_MAPKW_ASSIGN_SECTION      "ASSIGN_SECTION"
1541 @ MSG_MAPKW_AUX                 "AUXILIARY"
1542 @ MSG_MAPKW_CAPABILITY          "CAPABILITY"
1543 @ MSG_MAPKW_COMMON              "COMMON"
1544 @ MSG_MAPKW_DATA                "DATA"
1545 @ MSG_MAPKW_DEFAULT             "DEFAULT"
1546 @ MSG_MAPKW_DEPEND_VERSIONS     "DEPEND_VERSIONS"
1547 @ MSG_MAPKW_DIRECT              "DIRECT"
1548 @ MSG_MAPKW_DISABLE             "DISABLE"
1549 @ MSG_MAPKW_DYNSORT             "DYNSORT"
1550 @ MSG_MAPKW_ELIMINATE           "ELIMINATE"
1551 @ MSG_MAPKW_EXECUTE             "EXECUTE"
1552 @ MSG_MAPKW_EXPORTED            "EXPORTED"
1553 @ MSG_MAPKW_EXTERN              "EXTERN"
1554 @ MSG_MAPKW_FILTER              "FILTER"
1555 @ MSG_MAPKW_FILE_BASENAME       "FILE_BASENAME"
1556 @ MSG_MAPKW_FILE_PATH           "FILE_PATH"
1557 @ MSG_MAPKW_FILE_OBJNAME        "FILE_OBJNAME"
1558 @ MSG_MAPKW_FUNCTION            "FUNCTION"
1559 @ MSG_MAPKW_FLAGS               "FLAGS"
1560 @ MSG_MAPKW_GLOBAL              "GLOBAL"
1561 @ MSG_MAPKW_INTERPOSE           "INTERPOSE"
1562 @ MSG_MAPKW_HIDDEN              "HIDDEN"
1563 @ MSG_MAPKW_HDR_NOALLOC         "HDR_NOALLOC"
1564 @ MSG_MAPKW_HW                  "HW"
1565 @ MSG_MAPKW_HW_1                "HW_1"
1566 @ MSG_MAPKW_HW_2                "HW_2"
1567 @ MSG_MAPKW_IS_NAME             "IS_NAME"
1568 @ MSG_MAPKW_IS_ORDER            "IS_ORDER"
1569 @ MSG_MAPKW_LOAD_SEGMENT        "LOAD_SEGMENT"
1570 @ MSG_MAPKW_LOCAL               "LOCAL"
1571 @ MSG_MAPKW_MACHINE             "MACHINE"
1572 @ MSG_MAPKW_MAX_SIZE            "MAX_SIZE"
1573 @ MSG_MAPKW_NOHDR               "NOHDR"
1574 @ MSG_MAPKW_NODIRECT            "NODIRECT"
1575 @ MSG_MAPKW_NODYNSORT           "NODYNSORT"
1576 @ MSG_MAPKW_NOTE_SEGMENT        "NOTE_SEGMENT"
1577 @ MSG_MAPKW_NULL_SEGMENT        "NULL_SEGMENT"
1578 @ MSG_MAPKW_OS_ORDER            "OS_ORDER"
1579 @ MSG_MAPKW_PADDR               "PADDR"
```

```
1580 @ MSG_MAPKW_PARENT              "PARENT"
1581 @ MSG_MAPKW_PHDR_ADD_NULL       "PHDR_ADD_NULL"
1582 @ MSG_MAPKW_PLATFORM            "PLATFORM"
1583 @ MSG_MAPKW_PROTECTED           "PROTECTED"
1584 @ MSG_MAPKW_READ                "READ"
1585 @ MSG_MAPKW_ROUND               "ROUND"
1586 @ MSG_MAPKW_REQUIRE             "REQUIRE"
1587 @ MSG_MAPKW_SEGMENT_ORDER       "SEGMENT_ORDER"
1588 @ MSG_MAPKW_SF                  "SF"
1589 @ MSG_MAPKW_SF_1                "SF_1"
1590 @ MSG_MAPKW_SINGLETON           "SINGLETON"
1591 @ MSG_MAPKW_SIZE                "SIZE"
1592 @ MSG_MAPKW_SIZE_SYMBOL         "SIZE_SYMBOL"
1593 @ MSG_MAPKW_STACK               "STACK"
1594 @ MSG_MAPKW_SYMBOL_SCOPE        "SYMBOL_SCOPE"
1595 @ MSG_MAPKW_SYMBOL_VERSION      "SYMBOL_VERSION"
1596 @ MSG_MAPKW_SYMBOLIC            "SYMBOLIC"
1597 @ MSG_MAPKW_TYPE                "TYPE"
1598 @ MSG_MAPKW_VADDR               "VADDR"
1599 @ MSG_MAPKW_VALUE               "VALUE"
1600 @ MSG_MAPKW_WRITE               "WRITE"


1603 @ MSG_STR_DTRACE        "PT_SUNWDTRACE"
```

```
**********************************************************
   88579 Fri Nov 18 16:39:50 2016
new/usr/src/cmd/sgs/packages/common/SUNWonld-README
7594 ld -zaslr should accept Solaris-compatible values
**********************************************************
```

```
#
# 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
```

```
128 ------------------------------------------------------------------------------
129 4256518 miscalculated calloc() during dlclose/tsorting can result in segv
130 4254171 DT_SPARC_REGISTER has invalid value associated with it.
131 ------------------------------------------------------------------------------
132 All the above changes are incorporated in the following patches:
133         Solaris/SunOS 5.7_sparc         patch 106950-07
134         Solaris/SunOS 5.7_x86           patch 106951-07
135         Solaris/SunOS 5.6_sparc         patch 107733-04  (never released)
136         Solaris/SunOS 5.6_x86           patch 107734-04
137 ------------------------------------------------------------------------------
138 4293159 ld needs to combine sections with and without SHF_ORDERED flag(comdat)
139 4292238 linking a library which has a static char ptr invokes mprotect() call
140 ------------------------------------------------------------------------------
141 All the above changes except for:
142     4256518 miscalculated calloc() during dlclose/tsorting can result in segv
143     4254171 DT_SPARC_REGISTER has invalid value associated with it.
144 plus:
145     4238973 fix for 4121152 affects linking of Ada objects
146     4158744 patch 103627-02 causes core when RPATH has blank entry and
147             dlopen/dlclose is used
148 are incorporated in the following patches:
149         Solaris/SunOS 5.5.1_sparc       patch 103627-13
150         Solaris/SunOS 5.5.1_x86         patch 103628-12
151 ------------------------------------------------------------------------------
152 All the above changes are incorporated in the following patches:
153         Solaris/SunOS 5.7_sparc         patch 106950-08
154         Solaris/SunOS 5.7_x86           patch 106951-08
155         Solaris/SunOS 5.6_sparc         patch 107733-05
156         Solaris/SunOS 5.6_x86           patch 107734-05
157 ------------------------------------------------------------------------------
158 4295613 COMMON symbol resolution can be incorrect
159 ------------------------------------------------------------------------------
160 All the above changes plus:
161     4238973 fix for 4121152 affects linking of Ada objects
162     4158744 patch 103627-02 causes core when RPATH has blank entry and
163             dlopen/dlclose is used
164 are incorporated in the following patches:
165         Solaris/SunOS 5.5.1_sparc       patch 103627-14
166         Solaris/SunOS 5.5.1_x86         patch 103628-13
167 ------------------------------------------------------------------------------
168 All the above changes plus:
169     4351197 nfs performance problem by 103627-13
170 are incorporated in the following patches:
171         Solaris/SunOS 5.5.1_sparc       patch 103627-15
172         Solaris/SunOS 5.5.1_x86         patch 103628-14
173 ------------------------------------------------------------------------------
174 All the above changes are incorporated in the following patches:
175         Solaris/SunOS 5.7_sparc         patch 106950-09
176         Solaris/SunOS 5.7_x86           patch 106951-09
177         Solaris/SunOS 5.6_sparc         patch 107733-06
178         Solaris/SunOS 5.6_x86           patch 107734-06
179 ------------------------------------------------------------------------------
180 4158971 increase the default segment alignment for i386 to 64k
181 4064994 Add an $ISALIST token to those understood by the dynamic linker
182 xxxxxxx ia64 common code putback
183 4239308 LD_DEBUG busted for sparc machines
184 4239008 Support MAP_ANON
185 4238494 link-auditing extensions required
186 4232239 R_SPARC_LOX10 truncates field
187 4231722 R_SPARC_UA* relocations are busted
188 4235514 R_SPARC_OLO10 relocation fails
189 4244025 sgsmsg update
190 4239281 need to support SECREL relocations for ia64
191 4253751 ia64 linker must support PT_IA_64_UNWIND tables
192 4259254 dlmopen mistakenly closes fd 0 (stdin) under certain error conditions
193 4260872 libelf hangs when libthread present
```



```
194 4224569 linker core dumping when profiling specified
195 4270937 need mechanism to suppress ld.so.1's use of a default search path.
196 1050476 ld.so to permit configuration of search path
197 4273654 filtee processing using $ISALIST could be optimized
198 4271860 get MERCED cruft out of elf.h
199 4248991 Dynamic loader (via PLT) corrupts register G4
200 4275754 cannot mmap file: Resource temporarily unavailable
201 4277689 The linker can not handle relocation against MOVE tabl
202 4270766 atexit processing required on dlclose().
203 4279229 Add a "release" token to those understood by the dynamic linker
204 4215433 ld can bus error when insufficient disc space exists for output file
205 4285571 Pssst, want some free disk space?  ld's miscalculating.
206 4286236 ar gives confusing "bad format" error with a null .stab section
207 4286838 ld.so.1 can't handle a no-bits segment
208 4287364 ld.so.1 runtime configuration cleanup
209 4289573 disable linking of ia64 binaries for Solaris8
210 4293966 crle(1)'s default directories should be supplied
211 ------------------------------------------------------------------------------

213 ----------------------------------
214 Solaris 8 600 (1st Q-update - s28u1)
215 ----------------------------------
216 Bugid   Risk Synopsis
217 ==============================================================================
218 4309212 dlsym can't find symbol
219 4311226 rejection of preloading in secure apps is inconsistent
220 4312449 dlclose: invalid deletion of dependency can occur using RTLD_GLOBAL
221 ------------------------------------------------------------------------------
222 All the above changes are incorporated in the following patches:
223         Solaris/SunOS 5.8_sparc         patch 109147-01
224         Solaris/SunOS 5.8_x86           patch 109148-01
225         Solaris/SunOS 5.7_sparc         patch 106950-10
226         Solaris/SunOS 5.7_x86           patch 106951-10
227         Solaris/SunOS 5.6_sparc         patch 107733-07
228         Solaris/SunOS 5.6_x86           patch 107734-07
229 ------------------------------------------------------------------------------

231 ----------------------------------
232 Solaris 8 900 (2nd Q-update - s28u2)
233 ----------------------------------
234 Bugid   Risk Synopsis
235 ==============================================================================
236 4324775 non-PIC code & -zcombreloc don't mix very well...
237 4327653 run-time linker should preload tables it will process (madvise)
238 4324324 shared object code can be referenced before .init has fired
239 4321634 .init firing of multiple INITFIRST objects can fail
240 ------------------------------------------------------------------------------
241 All the above changes are incorporated in the following patches:
242         Solaris/SunOS 5.8_sparc         patch 109147-03
243         Solaris/SunOS 5.8_x86           patch 109148-03
244         Solaris/SunOS 5.7_sparc         patch 106950-11
245         Solaris/SunOS 5.7_x86           patch 106951-11
246         Solaris/SunOS 5.6_sparc         patch 107733-08
247         Solaris/SunOS 5.6_x86           patch 107734-08
248 ------------------------------------------------------------------------------
249 4338812 crle(1) omits entries in the directory cache
250 4341496 RFE: provide a static version of /usr/bin/crle
251 4340878 rtld should treat $ORIGIN like LD_LIBRARY_PATH in security issues
252 ------------------------------------------------------------------------------
253 All the above changes are incorporated in the following patches:
254         Solaris/SunOS 5.8_sparc         patch 109147-04
255         Solaris/SunOS 5.8_x86           patch 109148-04
256         Solaris/SunOS 5.7_sparc         patch 106950-12
257         Solaris/SunOS 5.7_x86           patch 106951-12
258 ------------------------------------------------------------------------------
259 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
```

```
524         Solaris/SunOS 5.6_x86           patch T107734-11
525 --------------------------------------------------------------------------------

527 ----------------------------------
528 Solaris 9 1202 (2nd Q-update - s9u2)
529 ----------------------------------
530 Bugid   Risk Synopsis
531 ================================================================================
532 4546416 add help messages to ld.so mdbmodule
533 4526752 we should build and ship ld.so's mdb module
534 4624658 update 386 TLS relocation values
535 4622472 LA_SYMB_DLSYM not set for la_symbind() invocations
536 4638070 ldd/ld.so.1 could aid in detecting unreferenced dependencies
537         PSARC/2002/096 Detecting unreferenced dependencies with ldd(1)
538 4633860 Optimization for unused static global variables
539         PSARC/2002/113 ld -zignore - section elimination
540 4642829 ld.so.1 mprotect()'s text segment for weak relocations (it shouldn't)
541 4621479 'make' in $SRC/cmd/sgs/tools tries to install things in the proto area
542 4529912 purge ia64 source from sgs
543 4651709 dlopen(RTLD_NOLOAD) can disable lazy loading
544 4655066 crle: -u with nonexistent config file doesn't work
545 4654406 string tables created by the link-editor could be smaller...
546         PSARC/2002/160 ld -znocompstrtab - disable string-table compression
547 4651493 RTLD_NOW can result in binding to an object prior to its init being run.
548 4662575 linker displacement relocation checking introduces significant
549         linker overhead
550 4533195 ld interposes on malloc()/free() preventing support library from freeing
551         memory
552 4630224 crle get's confused about memory layout of objects...
553 4664855 crle on application failed with ld.so.1 encountering mmap() returning
554         ENOMEM err
555 4669582 latest dynamic linker causes libthread _init to get skipped
556 4671493 ld.so.1 inconsistantly assigns PATHNAME() on primary objects
557 4668517 compile with map.bssalign doesn't copy _iob to bss
558 --------------------------------------------------------------------------------
559 All the above changes are incorporated in the following patches:
560         Solaris/SunOS 5.9_sparc         patch T112963-01
561         Solaris/SunOS 5.8_sparc         patch T109147-17
562         Solaris/SunOS 5.8_x86           patch T109148-17
563 --------------------------------------------------------------------------------
564 4701749 On Solaris 8 + 109147-16 ld crashes when building a dynamic library.
565 4707808 The ldd command is broken in the latest 2.8 linker patch.
566 --------------------------------------------------------------------------------
567 All the above changes are incorporated in the following patches:
568         Solaris/SunOS 5.9_sparc         patch T112963-02
569         Solaris/SunOS 5.8_sparc         patch T109147-18
570         Solaris/SunOS 5.8_x86           patch T109148-18
571 --------------------------------------------------------------------------------
572 4696204 enable extended section indexes in relocatable objects
573         PSARC/2001/332 ELF gABI updates - round II
574         PSARC/2002/369 libelf interfaces to support ELF Extended Sections
575 4706503 linkers need to cope with EF_SPARCV9_PSO/EF_SPARCV9_RMO
576 4716929 updating of local register symbols in dynamic symtab busted...
577 4710814 add "official" support for the "symbolic" keyword in linker map-file
578         PSARC/2002/439 linker mapfile visibility declarations
579 --------------------------------------------------------------------------------
580 All the above changes are incorporated in the following patches:
581         Solaris/SunOS 5.9_sparc         patch T112963-03
582         Solaris/SunOS 5.8_sparc         patch T109147-19
583         Solaris/SunOS 5.8_x86           patch T109148-19
584         Solaris/SunOS 5.7_sparc         patch T106950-19
585         Solaris/SunOS 5.7_x86           patch T106951-19
586 --------------------------------------------------------------------------------

588 ----------------------------------
589 Solaris 9 403 (3nd Q-update - s9u3)
```



```
590 ----------------------------------
591 Bugid   Risk Synopsis
592 ================================================================================
593 4731174 strip(1) does not fixup SHT_GROUP data
594 4733697 -zignore with gcc may exclude C++ exception sections
595 4733317 R_SPARC_*_HIX22 calculations are wrong with 32bit LD building
596         ELF64 binaries
597 4735165 fatal linker error when compiling C++ programs with -xlinkopt
598 4736951 The mcs broken when the target file is an archive file
599 --------------------------------------------------------------------------------
600 All the above changes are incorporated in the following patches:
601         Solaris/SunOS 5.8_sparc         patch T109147-20
602         Solaris/SunOS 5.8_x86           patch T109148-20
603         Solaris/SunOS 5.7_sparc         patch T106950-20
604         Solaris/SunOS 5.7_x86           patch T106951-20
605 --------------------------------------------------------------------------------
606 4739660 Threads deadlock in schedlock and dynamic linker lock.
607 4653148 ld.so.1/libc should unregister its dlclose() exit handler via a fini.
608 4743413 ld.so.1 doesn't terminate argv with NULL pointer when invoked directly
609 4746231 linker core-dumps when SECTION relocations are made against discarded
610         sections
611 4730433 ld.so.1 wastes time repeatedly opening dependencies
612 4744337 missing RD_CONSISTENT event with dlmopen(LD_ID_NEWLM, ...)
613 4670835 rd_load_objiter can ignore callback's return value
614 4745932 strip utility doesn't strip out Dwarf2 debug section
615 4754751 "strip" command doesn't remove comdat stab sections.
616 4755674 Patch 109147-18 results in coredump.
617 --------------------------------------------------------------------------------
618 All the above changes are incorporated in the following patches:
619         Solaris/SunOS 5.9_sparc         patch T112963-04
620         Solaris/SunOS 5.7_sparc         patch T106950-21
621         Solaris/SunOS 5.7_x86           patch T106951-21
622 --------------------------------------------------------------------------------
623 4772927 strip core dumps on an archive library
624 4774727 direct-bindings can fail against copy-reloc symbols
625 --------------------------------------------------------------------------------
626 All the above changes are incorporated in the following patches:
627         Solaris/SunOS 5.9_sparc         patch T112963-05
628         Solaris/SunOS 5.9_x86           patch T113986-01
629         Solaris/SunOS 5.8_sparc         patch T109147-21
630         Solaris/SunOS 5.8_x86           patch T109148-21
631         Solaris/SunOS 5.7_sparc         patch T106950-22
632         Solaris/SunOS 5.7_x86           patch T106951-22
633 --------------------------------------------------------------------------------

635 ----------------------------------
636 Solaris 9 803 (4th Q-update - s9u4)
637 ----------------------------------
638 Bugid   Risk Synopsis
639 ================================================================================
640 4730110 ld.so.1 list implementation could scale better
641 4728822 restrict the objects dlsym() searches.
642         PSARC/2002/478 New dlopen(3dl) flag - RTLD_FIRST
643 4714146 crle: 64-bit secure pathname is incorrect.
644 4504895 dlclose() does not remove all objects
645 4698800 Wrong comments in /usr/lib/ld/sparcv9/map.*
646 4745129 dldump is inconsistent with .dynamic processing errors.
647 4753066 LD_SIGNAL isn't very useful in a threaded environment
648         PSARC/2002/569 New dlinfo(3dl) flag - RTLD_DI_SIGNAL
649 4765536 crle: symbolic links can confuse alternative object configuration info
650 4766815 ld -r of object the TLS data fails
651 4770484 elfdump can not handle stripped archive file
652 4770494 The ld command gives improper error message handling broken archive
653 4775738 overwriting output relocation table when 'ld -zignore' is used
654 4778247 elfdump -e of core files fails
655 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
--------------------------------------------------------------------------------

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

```
788 Solaris 9 904 (7th Q-update - s9u7)
789 ----------------------------------
790 Bugid   Risk Synopsis
791 ============================================================================
792 4912214 Having multiple of libc.so.1 in a link map causes malloc() to fail
793 4526878 ld.so.1 should pass MAP_ALIGN flag to give kernel more flexibility
794 4930997 sgs bld_vernote.ksh script needs to be hardend...
795 4796286 ld.so.1: scenario for trouble?
796 4930985 clean up cruft under usr/src/cmd/sgs/tools
797 4933300 remove references to Ultra-1 in librtld_db demo
798 4936305 string table compression is much too slow...
799 4939626 SUNWonld internal package must be updated...
800 4939565 per-symbol filtering required
801 4948119 ld(1) -z loadfltr fails with per-symbol filtering
802 4948427 ld.so.1 gives fatal error when multiple RTLDINFO objects are loaded
803 4940894 ld core dumps using "-xldscope=symbolic
804 4955373 per-symbol filtering refinements
805 4878827 crle(1M) - display post-UPM search paths, and compensate for pre-UPM.
806 4955802 /usr/ccs/bin/ld dumps core in process_reld()
807 4964415 elfdump issues wrong relocation error message
808 4966465 LD_NOAUXFLTR fails when object is both a standard and auxiliary filter
809 4973865 the link-editor does not scale properly when linking objects with
810         lots of syms
811 4975598 SHT_SUNW_ANNOTATE section relocation not resolved
812 4974828 nss_files nss_compat r_mt tests randomly segfaulting
813 ----------------------------------------------------------------------------
814 All the above changes are incorporated in the following patches:
815         Solaris/SunOS 5.9_sparc         patch T112963-13
816         Solaris/SunOS 5.9_x86           patch T113986-09
817 ----------------------------------------------------------------------------
818 4860508 link-editors should create/promote/verify hardware capabilities
819 5002160 crle: reservation for dumped objects gets confused by mmaped object
820 4967869 linking stripped library causes segv in linker
821 5006657 link-editor doesn't always handle nodirect binding syminfo information
822 4915901 no way to see ELF information
823 5021773 ld.so.1 has trouble with objects having more than 2 segments.
824 ----------------------------------------------------------------------------
825 All the above changes are incorporated in the following patches:
826         Solaris/SunOS 5.9_sparc         patch T112963-14
827         Solaris/SunOS 5.9_x86           patch T113986-10
828         Solaris/SunOS 5.8_sparc         patch T109147-29
829         Solaris/SunOS 5.8_x86           patch T109148-29
830 ----------------------------------------------------------------------------
831 All the above changes plus:
832         6850124 dlopen reports "No such file or directory" in spite of ENOMEM
833                 when mmap fails in anon_map()
834 are incorporated in the following patches:
835         Solaris/SunOS 5.9_sparc         patch TXXXXXX-XX
836         Solaris/SunOS 5.9_x86           patch TXXXXXX-XX
837 ----------------------------------------------------------------------------

839 ----------
840 Solaris 10
841 ----------
842 Bugid   Risk Synopsis
843 ============================================================================
844 5044797 ld.so.1: secure directory testing is being skipped during filtee
845         processing
846 4963676 Remove remaining static libraries
847 5021541 unnecessary PT_SUNWBSS segment may be created
848 5031495 elfdump complains about bad symbol entries in core files
849 5012172 Need error when creating shared object with .o compiled
850         -xarch=v9 -xcode=abs44
851 4994738 rd_plt_resolution() resolves ebx-relative PLT entries incorrectly
852 5023493 ld -m output with patch 109147-25 missing .o information
853 ----------------------------------------------------------------------------
```



```
854 All the above changes are incorporated in the following patches:
855         Solaris/SunOS 5.9_sparc         patch T112963-15
856         Solaris/SunOS 5.9_x86           patch T113986-11
857         Solaris/SunOS 5.8_sparc         patch T109147-30
858         Solaris/SunOS 5.8_x86           patch T109148-30
859 ----------------------------------------------------------------------------
860 5071614 109147-29 & -30 break the build of on28-patch on Solaris 8 2/04
861 5029830 crle: provide for optional alternative dependencies.
862 5034652 ld.so.1 should save, and print, more error messages
863 5036561 ld.so.1 outputs non-fatal fatal message about auxiliary filter libraries
864 5042713 4866170 broke ld.so's ::setenv
865 5047082 ld can core dump on bad gcc objects
866 5047612 ld.so.1: secure pathname verification is flawed with filter use
867 5047235 elfdump can core dump printing PT_INTERP section
868 4798376 nits in demo code
869 5041446 gelf_update_*() functions inconsistently return NULL or 0
870 5032364 M_ID_TLSBSS and M_ID_UNKNOWN have the same value
871 4707030 Empty LD_PRELOAD_64 doesn't override LD_PRELOAD
872 4968618 symbolic linkage causes core dump
873 5062313 dladdr() can cause deadlock in MT apps.
874 5056867 $ISALIST/$HWCAP expansion should be more flexible.
875 4918303 0@0.so.1 should not use compiler-supplied crt*.o files
876 5058415 whocalls cannot take more than 10 arguments
877 5067518 The fix for 4918303 breaks the build if a new work space is used.
878 ----------------------------------------------------------------------------
879 All the above changes are incorporated in the following patches:
880         Solaris/SunOS 5.9_sparc         patch T112963-16
881         Solaris/SunOS 5.9_x86           patch T113986-12
882         Solaris/SunOS 5.8_sparc         patch T109147-31
883         Solaris/SunOS 5.8_x86           patch T109148-31
884 ----------------------------------------------------------------------------
885 5013759 *file* should report hardware/software capabilities (link-editor
886         components only)
887 5063580 libldstab: file /tmp/posto..: .stab[.index|.sbfocus] found with no
888         matching stri
889 5076838 elfdump(1) is built with a CTF section (the wrong one)
890 5080344 Hardware capabilities are not enforced for a.out
891 5079061 RTLD_DEFAULT can be expensive
892         PSARC/2004/747 New dlsym(3c) Handle - RTLD_PROBE
893 5064973 allow normal relocs against TLS symbols for some sections
894 5085792 LD_XXXX_64 should override LD_XXXX
895 5096272 every executable or library has a .SUNW_dof section
896 5094135 Bloomberg wants a faster ldd.
897 5086352 libld.so.3 should be built with a .SUNW_ctf ELF section, ready for CR
898 5098205 elfdump gives wrong section name for the global offset table
899 5092414 Linker patch 109147-29 makes Broadvison One-To-One server v4.1
900         installation fail
901 5080256 dump(1) doesn't list ELF hardware capabilities
902 5097347 recursive read lock in gelf_getsym()
903 ----------------------------------------------------------------------------
904 All the above changes are incorporated in the following patches:
905         Solaris/SunOS 5.9_sparc         patch T112963-17
906         Solaris/SunOS 5.9_x86           patch T113986-13
907         Solaris/SunOS 5.8_sparc         patch T109147-32
908         Solaris/SunOS 5.8_x86           patch T109148-32
909 ----------------------------------------------------------------------------
910 5106206 ld.so.1 fail to run a Solaris9 program that has libc linked with
911         -z lazyload
912 5102601 ON should deliver a 64-bit operating system for Opteron systems
913         (link-editor components only)
914 6173852 enable link_auditing technology for amd64
915 6174599 linker does not create .eh_frame_hdr sections for eh_frame sections
916         with SHF_LINK_ORDER
917 6175609 amd64 run-time linker has a corrupted note section
918 6175843 amd64 rdb_demo files not installed
919 6182293 ld.so.1 can repeatedly relocate object .plts (RTLD_NOW).
```

```
 920 6183645 ld core dumps when automounter fails
 921 6178667 ldd list unexpected (file not found) in x86 environment.
 922 6181928 Need new reloc types R_AMD64_GOTOFF64 and R_AMD64_GOTPC32
 923 6182884 AMD64: ld coredumps when building a shared library
 924 6173559 The ld may set incorrect value for sh_addralign under some conditions.
 925 5105601 ld.so.1 gets a little too enthusiastic with interposition
 926 6189384 ld.so.1 should accommodate a files dev/inode change (libc loopback mnt)
 927 6177838 AMD64: linker cannot resolve PLT for 32-bit a.out(s) on amd64-S2 kernel
 928 6190863 sparc disassembly code should be removed from rdb_demo
 929 6191488 unwind eh_frame_hdr needs corrected encoding value
 930 6192490 moe(1) returns /lib/libc.so.1 for optimal expansion of libc HWCAP
 931         libraries
 932 6192164 AMD64: introduce dlamd64getunwind interface
 933         PSARC/2004/747 libc::dlamd64getunwind()
 934 6195030 libdl has bad version name
 935 6195521 64-bit moe(1) missed the train
 936 6198358 AMD64: bad eh_frame_hdr data when C and C++ mixed in a.out
 937 6204123 ld.so.1: symbol lookup fails even after lazy loading fallback
 938 6207495 UNIX98/UNIX03 vsx namespace violation DYNL.hdr/misc/dlfcn/T.dlfcn
 939         14 Failed
 940 6217285 ctfmerge crashed during full onnv build
 941 --------------------------------------------------------------------------------

 943 ------------------------------------
 944 Solaris 10 106 (1st Q-update - s10u1)
 945 ------------------------------------
 946 Bugid   Risk Synopsis
 947 ================================================================================
 948 6209350 Do not include signature section from dynamic dependency library into
 949         relocatable object
 950 6212797 The binary compiled on SunOS4.x doesn't run on Solaris8 with Patch
 951         109147-31
 952 --------------------------------------------------------------------------------
 953 All the above changes are incorporated in the following patches:
 954         Solaris/SunOS 5.9_sparc         patch T112963-18
 955         Solaris/SunOS 5.9_x86           patch T113986-14
 956         Solaris/SunOS 5.8_sparc         patch T109147-33
 957         Solaris/SunOS 5.8_x86           patch T109148-33
 958 --------------------------------------------------------------------------------
 959 6219538 112963-17: linker patch causes binary to dump core
 960 --------------------------------------------------------------------------------
 961 All the above changes are incorporated in the following patches:
 962         Solaris/SunOS 5.10_sparc        patch T117461-01
 963         Solaris/SunOS 5.10_x86          patch T118345-01
 964         Solaris/SunOS 5.9_sparc         patch T112963-19
 965         Solaris/SunOS 5.9_x86           patch T113986-15
 966         Solaris/SunOS 5.8_sparc         patch T109147-34
 967         Solaris/SunOS 5.8_x86           patch T109148-34
 968 --------------------------------------------------------------------------------
 969 6257177 incremental builds of usr/src/cmd/sgs can fail...
 970 6219651 AMD64: Linker does not issue error for out of range R_AMD64_PC32
 971 --------------------------------------------------------------------------------
 972 All the above changes are incorporated in the following patches:
 973         Solaris/SunOS 5.10_sparc        patch T117461-02
 974         Solaris/SunOS 5.10_x86          patch T118345-02
 975         Solaris/SunOS 5.9_sparc         patch T112963-20
 976         Solaris/SunOS 5.9_x86           patch T113986-16
 977         Solaris/SunOS 5.8_sparc         patch T109147-35
 978         Solaris/SunOS 5.8_x86           patch T109148-35
 979 NOTE: The fix for 6219651 is only applicable for 5.10_x86 platform.
 980 --------------------------------------------------------------------------------
 981 5080443 lazy loading failure doesn't clean up after itself (D)
 982 6226206 ld.so.1 failure when processing single segment hwcap filtee
 983 6228472 ld.so.1: link-map control list stacking can loose objects
 984 6235000 random packages not getting installed in snv_09 and snv_10 -
 985         rtld/common/malloc.c Assertion
```



```
 986 6219317 Large page support is needed for mapping executables, libraries and
 987         files (link-editor components only)
 988 6244897 ld.so.1 can't run apps from commandline
 989 6251798 moe(1) returns an internal assertion failure message in some
 990         circumstances
 991 6251722 ld fails silently with exit 1 status when -z ignore passed
 992 6254364 ld won't build libgenunix.so with absolute relocations
 993 6215444 ld.so.1 caches "not there" lazy libraries, foils svc.startd(1M)'s logic
 994 6222525 dlsym(3C) trusts caller(), which may return wrong results with tail call
 995         optimization
 996 6241995 warnings in sgs should be fixed (link-editor components only)
 997 6258834 direct binding availability should be verified at runtime
 998 6260361 lari shouldn't count a.out non-zero undefined entries as interesting
 999 6260780 ldd doesn't recognize LD_NOAUXFLTR
1000 6266261 Add ld(1) -Bnodirect support (D)
1001 6261990 invalid e_flags error could be a little more friendly
1002 6261803 lari(1) should find more events uninteresting (D)
1003 6267352 libld_malloc provides inadequate alignment
1004 6268693 SHN_SUNW_IGNORE symbols should be allowed to be mulitiply defined
1005 6262789 Infosys wants a faster linker
1006 --------------------------------------------------------------------------------
1007 All the above changes are incorporated in the following patches:
1008         Solaris/SunOS 5.10_sparc        patch T117461-03
1009         Solaris/SunOS 5.10_x86          patch T118345-03
1010         Solaris/SunOS 5.9_sparc         patch T112963-21
1011         Solaris/SunOS 5.9_x86           patch T113986-17
1012         Solaris/SunOS 5.8_sparc         patch T109147-36
1013         Solaris/SunOS 5.8_x86           patch T109148-36
1014 --------------------------------------------------------------------------------
1015 6283601 The usr/src/cmd/sgs/packages/common/copyright contains old information
1016         legally problematic
1017 6276905 dlinfo gives inconsistent results (relative vs absolute linkname) (D)
1018         PSARC/2005/357 dlinfo(3c) RTLD_DI_ARGSINFO
1019 6284941 excessive link times with many groups/sections
1020 6280467 dlclose() unmaps shared library before library's _fini() has finished
1021 6291547 ld.so mishandles LD_AUDIT causing security problems.
1022 --------------------------------------------------------------------------------
1023 All the above changes are incorporated in the following patches:
1024         Solaris/SunOS 5.10_sparc        patch T117461-04
1025         Solaris/SunOS 5.10_x86          patch T118345-04
1026         Solaris/SunOS 5.9_sparc         patch T112963-22
1027         Solaris/SunOS 5.9_x86           patch T113986-18
1028         Solaris/SunOS 5.8_sparc         patch T109147-37
1029         Solaris/SunOS 5.8_x86           patch T109148-37
1030 --------------------------------------------------------------------------------
1031 6295971 UNIX98/UNIX03 *vsx* DYNL.hdr/misc/dlfcn/T.dlfcn 14 fails, auxv.h syntax
1032         error
1033 6299525 .init order failure when processing cycles
1034 6273855 gcc and sgs/crle don't get along
1035 6273864 gcc and sgs/libld don't get along
1036 6273875 gcc and sgs/rtld don't get along
1037 6272563 gcc and amd64/krtld/doreloc.c don't get along
1038 6290157 gcc and sgs/librtld_db/rdb_demo don't get along
1039 6301218 Matlab dumps core on startup when running on 112963-22 (D)
1040 --------------------------------------------------------------------------------
1041 All the above changes are incorporated in the following patches:
1042         Solaris/SunOS 5.10_sparc        patch T117461-06
1043         Solaris/SunOS 5.10_x86          patch T118345-08
1044         Solaris/SunOS 5.9_sparc         patch T112963-23
1045         Solaris/SunOS 5.9_x86           patch T113986-19
1046         Solaris/SunOS 5.8_sparc         patch T109147-38
1047         Solaris/SunOS 5.8_x86           patch T109148-38
1048 --------------------------------------------------------------------------------
1049 6314115 Checkpoint refuses to start, crashes on start, after application of
1050         linker patch 112963-22
1051 --------------------------------------------------------------------------------
```

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
```

```
1448 6681761 lies, darn lies, and linker README files
1449 6509323 Need to disable the Multiple Files loading - same name, different
1450         directories (or its stat() use)
1451 6686889 ld.so.1 regression - bad pointer created with 6509323 integration
1452 6695681 ldd(1) crashes when run from a chrooted environment
1453 6516212 usr/src/cmd/sgs/libelf warlock targets should be fixed or abandoned
1454 6678310 using LD_AUDIT, ld.so.1 calls shared library's .init before library is
1455         fully relocated (link-editor components only)
1456 6699594 The ld command has a problem handling 'protected' mapfile keyword.
1457 6699131 elfdump should display core file notes (D)
1458 6702260 single threading .init/.fini sections breaks staroffice
1459 6703919 boot hangs intermittently on x86 with onnv daily.0430 and on
1460 6701798 ld can enter infinite loop processing bad mapfile
1461 6706401 direct binding copy relocation fallback is insufficient for ild
1462         generated objects
1463 6705846 multithreaded C++ application seems to get deadlocked in the dynamic
1464         linker code
1465 6686343 ldd(1) - unused search path diagnosis should be enabled
1466 6712292 ld.so.1 should fall back to an interposer for failed direct bindings
1467 6716350 usr/src/cmd/sgs should be linted by nightly builds
1468 6720509 usr/src/cmd/sgs/sgsdemangler should be removed
1469 6617475 gas creates erroneous FILE symbols [was: ld.so.1 is reported as
1470         false positive by wsdiff]
1471 6724311 dldump() mishandles R_AMD64_JUMP_SLOT relocations
1472 6724774 elfdump -n doesn't print siginfo structure
1473 6728555 Fix for amd64 aw (6617475) breaks pure gcc builds
1474 6734598 ld(1) archive processing failure due to mismatched file descriptors (D)
1475 6735939 ld(1) discarded symbol relocations errors (Studio and GNU).
1476 6354160 Solaris linker includes more than one copy of code in binary when
1477         linking gnu object code
1478 6744003 ld(1) could provide better argument processing diagnostics (D)
1479         PSARC 2008/583 add gld options to ld(1)
1480 6749055 ld should generate GNU style VERSYM indexes for VERNEED records (D)
1481         PSARC/2008/603 ELF objects to adopt GNU-style Versym indexes
1482 6752728 link-editor can enter UNDEF symbols in symbol sort sections
1483 6756472 AOUT search path pruning (D)
1484 --------------------------------------------------------------------------------

1486 ------------------------------------------
1487 Solaris Nevada (OpenSolaris 2009.06, snv_111)
1488 ------------------------------------------
1489 Bugid   Risk Synopsis
1490 ================================================================================

1492 6754965 introduce the SF1_SUNW_ADDR32 bit in software capabilities (D)
1493         (link-editor components only)
1494         PSARC/2008/622 32-bit Address Restriction Software Capabilities Flag
1495 6756953 customer requests that DT_CONFIG strings be honored for secure apps (D)
1496 6765299 ld --version-script option not compatible with GNU ld (D)
1497 6748160 problem with -zrescan (D)
1498         PSARC/2008/651 New ld archive rescan options
1499 6763342 sloppy relocations need to get sloppier
1500 6736890 PT_SUNWBSS should be disabled (D)
1501         PSARC/2008/715 PT_SUNWBSS removal
1502 6772661 ldd/lddstub/ld.so.1 dump core in current nightly while processing
1503         libsoftcrypto_hwcap.so.1
1504 6765931 mcs generates unlink(NULL) system calls
1505 6775062 remove /usr/lib/libldstab.so (D)
1506 6782977 ld segfaults after support lib version error sends bad args to vprintf()
1507 6773695 ld -z nopartial can break non-pic objects
1508 6778453 RTLD_GROUP prevents use of application defined malloc
1509 6789925 64-bit applications with SF1_SUNW_ADDR32 require non-default starting
1510         address
1511 6792906 ld -z nopartial fix breaks TLS
1512 6686372 ld.so.1 should use mmapobj(2)
1513 6726108 dlopen() performance could be improved.
```



```
1514 6792836 ld is slow when processing GNU linkonce sections
1515 6797468 ld.so.1: orphaned handles aren't processed correctly
1516 6798676 ld.so.1: enters infinite loop with realloc/defragmentation logic
1517 6237063 request extension to dl* family to provide segment bounds
1518         information (D)
1519         PSARC/2009/054 dlinfo(3c) - segment mapping retrieval
1520 6800388 shstrtab can be sized incorrectly when -z ignore is used
1521 6805009 ld.so.1: link map control list tear down leaves dangling pointer -
1522         pfinstall does it again.
1523 6807050 GNU linkonce sections can create duplicate and incompatible
1524         eh_frame FDE entries
1525 --------------------------------------------------------------------------------

1527 --------------
1528 Solaris Nevada
1529 --------------
1530 Bugid   Risk Synopsis
1531 ================================================================================
1532 6813909 generalize eh_frame support to non-amd64 platforms
1533 6801536 ld: mapfile processing oddities unveiled through mmapobj(2) observations
1534 6802452 libelf shouldn't use MS_SYNC
1535 6818012 nm tries to modify readonly segment and dumps core
1536 6821646 xVM dom0 doesn't boot on daily.0324 and beyond
1537 6822828 librtld_db can return RD_ERR before RD_NOMAPS, which compromises dbx
1538         expectations.
1539 6821619 Solaris linkers need systematic approach to ELF OSABI (D)
1540         PSARC/2009/196 ELF objects to set OSABI / elfdump -O option
1541 6827468 6801536 breaks 'ld -s' if there are weak/strong symbol pairs
1542 6715578 AOUT (BCP) symbol lookup can be compromised with lazy loading.
1543 6752883 ld.so.1 error message should be buffered (not sent to stderr).
1544 6577982 ld.so.1 calls getpid() before it should when any LD_* are set
1545 6831285 linker LD_DEBUG support needs improvements (D)
1546 6806791 filter builds could be optimized (link-editor components only)
1547 6823371 calloc() uses suboptimal memset() causing 15% regression in SpecCPU2006
1548         gcc code (link-editor components only)
1549 6831308 ld.so.1: symbol rescanning does a little too much work
1550 6837777 ld ordered section code uses too much memory and works too hard
1551 6841199 Undo 10 year old workaround and use 64-bit ld on 32-bit objects
1552 6784790 ld should examine archives to determine output object class/machine (D)
1553         PSARC/2009/305 ld -32 option
1554 6849998 remove undocumented mapfile $SPECVERS and $NEED options
1555 6851224 elf_getshnum() and elf_getshstrndx() incompatible with 2002 ELF gABI
1556         agreement (D)
1557         PSARC/2009/363 replace elf_getphnum, elf_getshnum, and elf_getshstrndx
1558 6853809 ld.so.1: rescan fallback optimization is invalid
1559 6854158 ld.so.1: interposition can be skipped because of incorrect
1560         caller/destination validation
1561 6862967 rd_loadobj_iter() failing for core files
1562 6856173 streams core dumps when compiled in 64bit with a very large static
1563         array size
1564 6834197 ld pukes when given an empty plate
1565 6516644 per-symbol filtering shouldn't be allowed in executables
1566 6878605 ld should accept '%' syntax when matching input SHT_PROGBITS sections
1567 6850768 ld option to autogenerate wrappers/interposers similar to GNU ld
1568         --wrap (D)
1569         PSARC/2009/493 ld -z wrap option
1570 6888489 Null environment variables are not overriding crle(1) replaceable
1571         environment variables.
1572 6885456 Need to implement GNU-ld behavior in construction of .init/.fini
1573         sections
1574 6900241 ld should track SHT_GROUP sections by symbol name, not section name
1575 6901773 Special handling of STT_SECTION group signature symbol for GNU objects
1576 6901895 Failing asserts in ld update_osym() trying to build gcc 4.5 develpment
1577         head
1578 6909523 core dump when run "LD_DEBUG=help ls" in non-English locale
1579 6903688 mdb(1) can't resolve certain symbols in solaris10-branded processes
```

```
1580         from the global zone
1581 6923449 elfdump misinterprets _init/_fini symbols in dynamic section test
1582 6914728 Add dl_iterate_phdr() function to ld.so.1 (D)
1583         PSARC/2010/015 dl_iterate_phdr
1584 6916788 ld version 2 mapfile syntax (D)
1585         PSARC/2009/688 Human readable and extensible ld mapfile syntax
1586 6929607 ld generates incorrect VERDEF entries for ET_REL output objects
1587 6924224 linker should ignore SUNW_dof when calculating the elf checksum
1588 6918143 symbol capabilities (D)
1589         PSARC/2010/022 Linker-editors: Symbol Capabilities
1590 6910387 .tdata and .tbss separation invalidates TLS program header information
1591 6934123 elfdump -d coredumps on PA-RISC elf
1592 6931044 ld should not allow SHT_PROGBITS .eh_frame sections on amd64 (D)
1593 6931056 pvs -r output can include empty versions in output
1594 6938628 ld.so.1 should produce diagnostics for all dl*() entry points
1595 6938111 nm 'No symbol table data' message goes to stdout
1596 6941727 ld relocation cache memory use is excessive
1597 6932220 ld -z allextract skips objects that lack global symbols
1598 6943772 Testing for a symbols existence with RTLD_PROBE is compromised by
1599         RTLD_BIND_NOW
1600         PSARC/2010/XXX Deferred symbol references
1601 6943432 dlsym(RTLD_PROBE) should only bind to symbol definitions
1602 6668759 an external method for determining whether an ELF dependency is optional
1603 6954032 Support library with ld_open and -z allextract in snv_139 do not mix
1604 6949596 wrong section alignment generated in joint compilation with shared
1605         library
1606 6961755 ld.so.1's -e arguments should take precedence over environment
1607         variables. (D)
1608 6748925 moe returns wrong hwcap library in some circumstances
1609 6916796 OSnet mapfiles should use version 2 link-editor syntax
1610 6964517 OSnet mapfiles should use version 2 link-editor syntax (2nd pass)
1611 6948720 SHT_INIT_ARRAY etc. section names don't follow ELF gABI (D)
1612 6962343 sgsmsg should use mkstemp() for temporary file creation
1613 6965723 libsoftcrypto symbol capabilities rely on compiler generated
1614         capabilities - gcc failure (link-editor components only)
1615 6952219 ld support for archives larger than 2 GB (D, P)
1616         PSARC/2010/224 Support for archives larger than 2 GB
1617 6956152 dlclose() from an auditor can be fatal. Preinit/activity events should
1618         be more flexible. (D)
1619 6971440 moe can core dump while processing libc.
1620 6972234 sgs demo's could use some cleanup
1621 6935867 .dynamic could be readonly in sharable objects
1622 6975290 ld mishandles GOT relocation against local ABS symbol
1623 6972860 ld should provide user guidance to improve objects (D)
1624         PSARC/2010/312 Link-editor guidance
1625 ------------------------------------------------------------------------------

1627 --------------
1628 Illumos
1629 --------------
1630 Bugid   Risk Synopsis
1631 ==============================================================================

1633 308     ld may misalign sections only preceded by empty sections
1634 1301    ld crashes with '-z ignore' due to a null data descriptor
1635 1626    libld may accidentally return success while failing
1636 2413    %ymm* need to be preserved on way through PLT
1637 3210    ld should tolerate SHT_PROGBITS for .eh_frame sections on amd64
1638 3228    Want -zassert-deflib for ld
1639 3230    ld.so.1 should check default paths for DT_DEPAUDIT
1640 3260    linker is insufficiently careful with strtok
1641 3261    linker should ignore unknown hardware capabilities
1642 3265    link-editor builds bogus .eh_frame_hdr on ia32
1643 3453    GNU comdat redirection does exactly the wrong thing
1644 3439    discarded sections shouldn't end up on output lists
1645 3436    relocatable objects also need sloppy relocation
```



```
1646 3451    archive libraries with no symbols shouldn't require a string table
1647 3616    SHF_GROUP sections should not be discarded via other COMDAT mechanisms
1648 3709    need sloppy relocation for GNU .debug_macro
1649 3722    link-editor is over restrictive of R_AMD64_32 addends
1650 3926    multiple extern map file definitions corrupt symbol table entry
1651 3999    libld extended section handling is broken
1652 4003    dldump() can't deal with extended sections
1653 4227    ld --library-path is translated to -l-path, not -L
1654 4270    ld(1) argument error reporting is still pretty bad
1655 4383    libelf can't write extended sections when ELF_F_LAYOUT
1656 4959    completely discarded merged string sections will corrupt output objects
1657 4996    rtld _init race leads to incorrect symbol values
1658 5688    ELF tools need to be more careful with dwarf data
1659 6098    ld(1) should not require symbols which identify group sections be global
1660 6252    ld should merge function/data-sections in the same manner as GNU ld
1661 7323    ld(1) -zignore can erroneously discard init and fini arrays as unreferen
1662 7594    ld -zaslr should accept Solaris-compatible values
1663 #endif /* ! codereview */
```