```
**********************************************************
   42259 Thu Aug 25 11:51:57 2016
new/usr/src/cmd/sgs/libld/common/place.c
7323 ld(1) -zignore can erroneously discard init and fini arrays as unreferenced
**********************************************************
______unchanged_portion_omitted_

 715 /*
 716  * Place a section into the appropriate segment and output section.
 717  *
 718  * entry:
 719  *      ofl - File descriptor
 720  *      isp - Input section descriptor of section to be placed.
 721  *      path_info - NULL, or pointer to Place_path_info buffer initialized
 722  *              by ld_place_path_info_init() for the file associated to isp,
 723  *              for use in processing entrance criteria with non-empty
 724  *              file matching string list (ec_files)
 725  *      ident - Section identifier, used to order sections relative to
 726  *              others within the output segment.
 727  *      alt_os_name - If non-NULL, the name of the output section to place
 728  *              isp into. If NULL, input sections go to an output section
 729  *              with the same name as the input section.
 730  */
 731 Os_desc *
 732 ld_place_section(Ofl_desc *ofl, Is_desc *isp, Place_path_info *path_info,
 733     int ident, const char *alt_os_name)
 734 {
 735         Ent_desc        *enp;
 736         Sg_desc         *sgp;
 737         Os_desc         *osp;
 738         Aliste          idx1, iidx;
 739         int             os_ndx;
 740         Shdr            *shdr = isp->is_shdr;
 741         Xword           shflagmask, shflags = shdr->sh_flags;
 742         Ifl_desc        *ifl = isp->is_file;
 743         char            *oname, *sname;
 744         uint_t          onamehash;
 745         Boolean         is_ehframe = (isp->is_flags & FLG_IS_EHFRAME) != 0;

 747         /*
 748          * Define any sections that must be thought of as referenced.  These
 749          * sections may not be referenced externally in a manner ld(1) can
 750          * discover, but they must be retained (ie. not removed by -zignore).
 751          */
 752         static const Msg RefSecs[] = {
 753                 MSG_SCN_INIT,           /* MSG_ORIG(MSG_SCN_INIT) */
 754                 MSG_SCN_FINI,           /* MSG_ORIG(MSG_SCN_FINI) */
 755                 MSG_SCN_EX_RANGES,      /* MSG_ORIG(MSG_SCN_EX_RANGES) */
 756                 MSG_SCN_EX_SHARED,      /* MSG_ORIG(MSG_SCN_EX_SHARED) */
 757                 MSG_SCN_CTORS,          /* MSG_ORIG(MSG_SCN_CTORS) */
 758                 MSG_SCN_DTORS,          /* MSG_ORIG(MSG_SCN_DTORS) */
 759                 MSG_SCN_EHFRAME,        /* MSG_ORIG(MSG_SCN_EHFRAME) */
 760                 MSG_SCN_EHFRAME_HDR,    /* MSG_ORIG(MSG_SCN_EHFRAME_HDR) */
 761                 MSG_SCN_JCR,            /* MSG_ORIG(MSG_SCN_JCR) */
 762                 MSG_SCN_INITARRAY,      /* MSG_ORIG(MSG_SCN_INITARRAY) */
 763                 MSG_SCN_FINIARRAY,      /* MSG_ORIG(MSG_SCN_FINIARRAY) */
 764                 MSG_SCN_PREINITARRAY,   /* MSG_ORIG(MSG_SCN_PREINITARRAY) */
 765 #endif /* ! codereview */
 766                 0
 767         };

 769         DBG_CALL(Dbg_sec_in(ofl->ofl_lml, isp));

 771         /*
 772          * If this section identifies group members, or this section indicates
 773          * that it is a member of a group, determine whether the section is
```



```
 774          * still required.
 775          */
 776         if ((shflags & SHF_GROUP) || (shdr->sh_type == SHT_GROUP)) {
 777                 Group_desc      *gdesc;

 779                 if ((gdesc = ld_get_group(ofl, isp)) != NULL) {
 780                         DBG_CALL(Dbg_sec_group(ofl->ofl_lml, isp, gdesc));

 782                         /*
 783                          * If this group has been replaced by another group,
 784                          * then this section needs to be discarded.
 785                          */
 786                         if (gdesc->gd_oisc) {
 787                                 isp->is_flags |= FLG_IS_DISCARD;

 789                                 /*
 790                                  * Since we're discarding the section, we
 791                                  * can skip assigning it to an output section.
 792                                  * The exception is that if the user
 793                                  * specifies -z relaxreloc, then
 794                                  * we need to assign the output section so
 795                                  * that the sloppy relocation logic will have
 796                                  * the information necessary to do its work.
 797                                  */
 798                                 if (!(ofl->ofl_flags1 & FLG_OF1_RLXREL))
 799                                         return (NULL);
 800                         }
 801                 }

 803                 /*
 804                  * SHT_GROUP sections can only be included into relocatable
 805                  * objects.
 806                  */
 807                 if (shdr->sh_type == SHT_GROUP) {
 808                         if ((ofl->ofl_flags & FLG_OF_RELOBJ) == 0) {
 809                                 isp->is_flags |= FLG_IS_DISCARD;
 810                                 return (NULL);
 811                         }
 812                 }
 813         }

 815         /*
 816          * Always assign SHF_TLS sections to the DATA segment (and then the
 817          * PT_TLS embedded inside of there).
 818          */
 819         if (shflags & SHF_TLS)
 820                 shflags |= SHF_WRITE;

 822         /*
 823          * Traverse the entrance criteria list searching for a segment that
 824          * matches the input section we have.  If an entrance criterion is set
 825          * then there must be an exact match.  If we complete the loop without
 826          * finding a segment, then sgp will be NULL.
 827          */
 828         sgp = NULL;
 829         for (APLIST_TRAVERSE(ofl->ofl_ents, idx1, enp)) {

 831                 /* Disabled segments are not available for assignment */
 832                 if (enp->ec_segment->sg_flags & FLG_SG_DISABLED)
 833                         continue;

 835                 /*
 836                  * If an entrance criteria doesn't have any of its fields
 837                  * set, it will match any section it is tested against.
 838                  * We set the FLG_EC_CATCHALL flag on these, primarily because
 839                  * it helps readers of our debug output to understand what
```

```
840                  * the criteria means --- otherwise the user would just see
841                  * that every field is 0, but might not understand the
842                  * significance of that.
843                  *
844                  * Given that we set this flag, we can use it here as an
845                  * optimization to short circuit all of the tests in this
846                  * loop. Note however, that if we did not do this, the end
847                  * result would be the same --- the empty criteria will sail
848                  * past the following tests and reach the end of the loop.
849                  */
850                 if (enp->ec_flags & FLG_EC_CATCHALL) {
851                         sgp = enp->ec_segment;
852                         break;
853                 }
854
855                 if (enp->ec_type && (enp->ec_type != shdr->sh_type))
856                         continue;
857                 if (enp->ec_attrmask &&
858                     /* LINTED */
859                     (enp->ec_attrmask & enp->ec_attrbits) !=
860                     (enp->ec_attrmask & shflags))
861                         continue;
862                 if (enp->ec_is_name &&
863                     (strcmp(enp->ec_is_name, isp->is_name) != 0))
864                         continue;
865
866                 if ((alist_nitems(enp->ec_files) > 0) &&
867                     ((path_info == NULL) || !eval_ec_files(path_info, enp)))
868                         continue;
869
870                 /* All entrance criteria tests passed */
871                 sgp = enp->ec_segment;
872                 break;
873         }
874
875         /*
876          * The final entrance criteria record is a FLG_EC_CATCHALL that points
877          * at the final predefined segment "extra", and this final segment is
878          * tagged FLG_SG_NODISABLE. Therefore, the above loop must always find
879          * a segment.
880          */
881         assert(sgp != NULL);
882
883         /*
884          * Transfer the input section sorting key from the entrance criteria
885          * to the input section. A non-zero value means that the section
886          * will be sorted on this key amoung the other sections that have a
887          * non-zero key. These sorted sections are collectively placed at the
888          * head of the output section.
889          *
890          * If the sort key is 0, the section is placed after the sorted
891          * sections in the order they are encountered.
892          */
893         isp->is_ordndx = enp->ec_ordndx;
894
895         /* Remember that this entrance criteria has placed a section */
896         enp->ec_flags |= FLG_EC_USED;
897
898         /*
899          * If our caller has supplied an alternative name for the output
900          * section, then we defer to their request. Otherwise, the default
901          * is to use the same name as that of the input section being placed.
902          *
903          * The COMDAT, SHT_GROUP and GNU name translations that follow have
904          * the potential to alter this initial name.
905          */
```



```
906         oname = (char *)((alt_os_name == NULL) ? isp->is_name : alt_os_name);
907
908         /*
909          * Solaris section names may follow the convention:
910          *
911          *      section-name%symbol-name
912          *
913          * This convention has been used to order the layout of sections within
914          * segments for objects built with the compilers -xF option.  However,
915          * the final object should not contain individual section headers for
916          * all such input sections, instead the symbol name is stripped from the
917          * name to establish the final output section name.
918          *
919          * This convention has also been followed for COMDAT and sections
920          * identified though SHT_GROUP data.
921          *
922          * Strip out the % from the section name for:
923          *      - Non-relocatable objects
924          *      - Relocatable objects if input section sorting is
925          *        in force for the segment in question.
926          */
927         if (((ofl->ofl_flags & FLG_OF_RELOBJ) == 0) ||
928             (sgp->sg_flags & FLG_SG_IS_ORDER)) {
929                 if ((sname = strchr(isp->is_name, '%')) != NULL) {
930                         size_t  size = sname - isp->is_name;
931
932                         if ((oname = libld_malloc(size + 1)) == NULL)
933                                 return ((Os_desc *)S_ERROR);
934                         (void) strncpy(oname, isp->is_name, size);
935                         oname[size] = '\0';
936                         DBG_CALL(Dbg_sec_redirected(ofl->ofl_lml, isp, oname));
937                 }
938         }
939
940         /*
941          * When building relocatable objects, we must not redirect COMDAT
942          * section names into their outputs, such that our output object may
943          * be successfully used as an input object also requiring COMDAT
944          * processing
945          */
946
947         /*
948          * GNU section names may follow the convention:
949          *
950          *      .gnu.linkonce.*
951          *
952          * The .gnu.linkonce is a section naming convention that indicates a
953          * COMDAT requirement.  Determine whether this section follows the GNU
954          * pattern, and if so, determine whether this section should be
955          * discarded or retained.  The comparison of is_name[1] with 'g'
956          * is an optimization to skip using strncmp() too much. This is safe,
957          * because we know the name is not NULL, and therefore must have
958          * at least one character plus a NULL termination.
959          */
960         if ((isp->is_name == oname) && (isp->is_name[1] == 'g') &&
961             (strncmp(MSG_ORIG(MSG_SCN_GNU_LINKONCE), isp->is_name,
962             MSG_SCN_GNU_LINKONCE_SIZE) == 0)) {
963                 if ((ofl->ofl_flags & FLG_OF_RELOBJ) == 0) {
964                         if ((oname = (char *)gnu_linkonce_sec(isp->is_name)) !=
965                             isp->is_name) {
966                                 DBG_CALL(Dbg_sec_redirected(ofl->ofl_lml, isp,
967                                     oname));
968                         }
969                 }
970
971                 /*
```

```
 972                 * Explicitly identify this section type as COMDAT.  Also,
 973                 * enable relaxed relocation processing, as this is typically
 974                 * a requirement with .gnu.linkonce sections.
 975                 */
 976                isp->is_flags |= FLG_IS_COMDAT;
 977                if ((ofl->ofl_flags1 & FLG_OF1_NRLXREL) == 0)
 978                        ofl->ofl_flags1 |= FLG_OF1_RLXREL;
 979                DBG_CALL(Dbg_sec_gnu_comdat(ofl->ofl_lml, isp, TRUE,
 980                    (ofl->ofl_flags1 & FLG_OF1_RLXREL) != 0));
 981        }

 983        /*
 984         * GNU section names may also follow the convention:
 985         *
 986         *      section-name.symbol-name
 987         *
 988         * This convention is used when defining SHT_GROUP sections of type
 989         * COMDAT.  Thus, any group processing will have discovered any group
 990         * sections, and this identification can be triggered by a pattern
 991         * match section names.
 992         */
 993        if ((isp->is_name == oname) && (isp->is_flags & FLG_IS_COMDAT) &&
 994            ((sname = gnu_comdat_sym(ifl, isp)) != NULL)) {
 995                size_t  size = sname - isp->is_name;

 997                if ((ofl->ofl_flags & FLG_OF_RELOBJ) == 0) {
 998                        if ((oname = libld_malloc(size + 1)) == NULL)
 999                                return ((Os_desc *)S_ERROR);
1000                        (void) strncpy(oname, isp->is_name, size);
1001                        oname[size] = '\0';
1002                        DBG_CALL(Dbg_sec_redirected(ofl->ofl_lml, isp, oname));
1003                }

1005                /*
1006                 * Enable relaxed relocation processing, as this is
1007                 * typically a requirement with GNU COMDAT sections.
1008                 */
1009                if ((ofl->ofl_flags1 & FLG_OF1_NRLXREL) == 0) {
1010                        ofl->ofl_flags1 |= FLG_OF1_RLXREL;
1011                        DBG_CALL(Dbg_sec_gnu_comdat(ofl->ofl_lml, isp,
1012                            FALSE, TRUE));
1013                }
1014        }

1016        /*
1017         * GNU section names named section-name.symbol-name which are not
1018         * members of COMDAT groups are merged according to the behaviour of
1019         * the GNU link-editor.
1020         *
1021         * See the description of gnu_split_sec().
1022         */
1023        if (((ofl->ofl_flags & FLG_OF_RELOBJ) == 0) &&
1024            (isp->is_name == oname) &&
1025            ((oname = (char *)gnu_split_sec(oname)) != isp->is_name)) {
1026                DBG_CALL(Dbg_sec_redirected(ofl->ofl_lml, isp, oname));
1027        }

1029        /*
1030         * Assign a hash value now that the output section name has been
1031         * finalized.
1032         */
1033        onamehash = sgs_str_hash(oname);

1035        /*
1036         * Determine if output section ordering is turned on. If so, return
1037         * the appropriate ordering index for the section. This information
```



```
1038         * is derived from the Sg_desc->sg_os_order list that was built
1039         * up from the Mapfile.
1040         *
1041         * A value of 0 for os_ndx means that the section is not sorted
1042         * (i.e. is not found in the sg_os_order). The items in sg_os_order
1043         * are in the desired sort order, so adding 1 to their alist index
1044         * gives a suitable index for sorting.
1045         */
1046        os_ndx = 0;
1047        if (alist_nitems(sgp->sg_os_order) > 0) {
1048                Sec_order       *scop;

1050                for (ALIST_TRAVERSE(sgp->sg_os_order, idx1, scop)) {
1051                        if (strcmp(scop->sco_secname, oname) == 0) {
1052                                scop->sco_flags |= FLG_SGO_USED;
1053                                os_ndx = idx1 + 1;
1054                                break;
1055                        }
1056                }
1057        }

1059        /*
1060         * Mask of section header flags to ignore when matching sections. We
1061         * are more strict with relocatable objects, ignoring only the order
1062         * flags, and keeping sections apart if they differ otherwise. This
1063         * follows the policy that sections in a relative object should only
1064         * be merged if their flags are the same, and avoids destroying
1065         * information prematurely. For final products however, we ignore all
1066         * flags that do not prevent a merge.
1067         */
1068        shflagmask =
1069            (ofl->ofl_flags & FLG_OF_RELOBJ) ? ALL_SHF_ORDER : ALL_SHF_IGNORE;

1071        /*
1072         * Traverse the input section list for the output section we have been
1073         * assigned. If we find a matching section simply add this new section.
1074         */
1075        iidx = 0;
1076        for (APLIST_TRAVERSE(sgp->sg_osdescs, idx1, osp)) {
1077                Shdr    *os_shdr = osp->os_shdr;

1079                /*
1080                 * An input section matches an output section if:
1081                 * -    The ident values match
1082                 * -    The names match
1083                 * -    Not a GROUP section
1084                 * -    Not a DTrace dof section
1085                 * -    Section types match
1086                 * -    Matching section flags, after screening out the
1087                 *      shflagmask flags.
1088                 *
1089                 * Section types are considered to match if any one of
1090                 * the following are true:
1091                 * -    The type codes are the same
1092                 * -    Both are .eh_frame sections (regardless of type code)
1093                 * -    The input section is COMDAT, and the output section
1094                 *      is SHT_PROGBITS.
1095                 */
1096                if ((ident == osp->os_identndx) &&
1097                    (ident != ld_targ.t_id.id_rel) &&
1098                    (onamehash == osp->os_namehash) &&
1099                    (shdr->sh_type != SHT_GROUP) &&
1100                    (shdr->sh_type != SHT_SUNW_dof) &&
1101                    ((shdr->sh_type == os_shdr->sh_type) ||
1102                    (is_ehframe && (osp->os_flags & FLG_OS_EHFRAME)) ||
1103                    ((shdr->sh_type == SHT_SUNW_COMDAT) &&
```

```
1104                    (os_shdr->sh_type == SHT_PROGBITS))) &&
1105                    ((shflags & ~shflagmask) ==
1106                    (os_shdr->sh_flags & ~shflagmask)) &&
1107                    (strcmp(oname, osp->os_name) == 0)) {
1108                        uintptr_t       err;
1109
1110                        /*
1111                         * Process any COMDAT section, keeping the first and
1112                         * discarding all others.
1113                         */
1114                        if ((isp->is_flags & FLG_IS_COMDAT) &&
1115                            ((err = add_comdat(ofl, osp, isp)) != 1))
1116                                return ((Os_desc *)err);
1117
1118                        /*
1119                         * Set alignment
1120                         */
1121                        set_addralign(ofl, osp, isp);
1122
1123                        /*
1124                         * If this section is a non-empty TLS section indicate
1125                         * that a PT_TLS program header is required.
1126                         */
1127                        if ((shflags & SHF_TLS) && shdr->sh_size &&
1128                            ((ofl->ofl_flags & FLG_OF_RELOBJ) == 0))
1129                                ofl->ofl_flags |= FLG_OF_TLSPHDR;
1130
1131                        /*
1132                         * Insert the input section descriptor on the proper
1133                         * output section descriptor list.
1134                         *
1135                         * If this segment requires input section ordering,
1136                         * honor any mapfile specified ordering for otherwise
1137                         * unordered sections by setting the mapfile_sort
1138                         * argument of os_attach_isp() to True.
1139                         */
1140
1141                        if (os_attach_isp(ofl, osp, isp,
1142                            (sgp->sg_flags & FLG_SG_IS_ORDER) != 0) == 0)
1143                                return ((Os_desc *)S_ERROR);
1144
1145                        /*
1146                         * If this input section and file is associated to an
1147                         * artificially referenced output section, make sure
1148                         * they are marked as referenced also. This ensures
1149                         * that this input section and file isn't eliminated
1150                         * when -zignore is in effect.
1151                         *
1152                         * See -zignore comments when creating a new output
1153                         * section below.
1154                         */
1155                        if (((ifl &&
1156                            (ifl->ifl_flags & FLG_IF_IGNORE)) || DBG_ENABLED) &&
1157                            (osp->os_flags & FLG_OS_SECTREF)) {
1158                                isp->is_flags |= FLG_IS_SECTREF;
1159                                if (ifl)
1160                                        ifl->ifl_flags |= FLG_IF_FILEREF;
1161                        }
1162
1163                        DBG_CALL(Dbg_sec_added(ofl->ofl_lml, osp, sgp));
1164                        return (osp);
1165                }
1166
1167                /*
1168                 * Do we need to worry about section ordering?
1169                 */
```



```
1170                if (os_ndx) {
1171                        if (osp->os_ordndx) {
1172                                if (os_ndx < osp->os_ordndx)
1173                                        /* insert section here. */
1174                                        break;
1175                                else {
1176                                        iidx = idx1 + 1;
1177                                        continue;
1178                                }
1179                        } else {
1180                                /* insert section here. */
1181                                break;
1182                        }
1183                } else if (osp->os_ordndx) {
1184                        iidx = idx1 + 1;
1185                        continue;
1186                }
1187
1188                /*
1189                 * If the new sections identifier is less than that of the
1190                 * present input section we need to insert the new section
1191                 * at this point.
1192                 */
1193                if (ident < osp->os_identndx)
1194                        break;
1195
1196                iidx = idx1 + 1;
1197        }
1198
1199        /*
1200         * We are adding a new output section.  Update the section header
1201         * count and associated string size.
1202         *
1203         * If the input section triggering this output section has been marked
1204         * for discard, and if no other non-discarded input section comes along
1205         * to join it, then we will over count. We cannot know if this will
1206         * happen or not until all input is seen. Set FLG_OF_AJDOSCNT to
1207         * trigger a final count readjustment.
1208         */
1209        if (isp->is_flags & FLG_IS_DISCARD)
1210                ofl->ofl_flags |= FLG_OF_ADJOSCNT;
1211        ofl->ofl_shdrcnt++;
1212        if (st_insert(ofl->ofl_shdrsttab, oname) == -1)
1213                return ((Os_desc *)S_ERROR);
1214
1215        /*
1216         * Create a new output section descriptor.
1217         */
1218        if ((osp = libld_calloc(sizeof (Os_desc), 1)) == NULL)
1219                return ((Os_desc *)S_ERROR);
1220        if ((osp->os_shdr = libld_calloc(sizeof (Shdr), 1)) == NULL)
1221                return ((Os_desc *)S_ERROR);
1222
1223        /*
1224         * Convert COMDAT section to PROGBITS as this the first section of the
1225         * output section.  Save any COMDAT section for later processing, as
1226         * additional COMDAT sections that match this section need discarding.
1227         */
1228        if ((shdr->sh_type == SHT_SUNW_COMDAT) &&
1229            ((shdr = isp_convert_type(isp, SHT_PROGBITS)) == NULL))
1230                return ((Os_desc *)S_ERROR);
1231        if ((isp->is_flags & FLG_IS_COMDAT) &&
1232            (add_comdat(ofl, osp, isp) == S_ERROR))
1233                return ((Os_desc *)S_ERROR);
1234
1235        if (is_ehframe) {
```

```
1236                 /*
1237                  * Executable or sharable objects can have at most a single
1238                  * .eh_frame section. Detect attempts to create more than
1239                  * one. This occurs if the input sections have incompatible
1240                  * attributes.
1241                  */
1242                 if ((ofl->ofl_flags & FLG_OF_EHFRAME) &&
1243                     !(ofl->ofl_flags & FLG_OF_RELOBJ)) {
1244                         eh_frame_muldef(ofl, isp);
1245                         return ((Os_desc *)S_ERROR);
1246                 }
1247                 ofl->ofl_flags |= FLG_OF_EHFRAME;

1249                 /*
1250                  * For .eh_frame sections, we always set the type to be the
1251                  * type specified by the ABI.  This allows .eh_frame sections
1252                  * of type SHT_PROGBITS to be correctly merged with .eh_frame
1253                  * sections of the ABI-defined type (e.g. SHT_AMD64_UNWIND),
1254                  * with the output being of the ABI-defined type.
1255                  */
1256                 osp->os_shdr->sh_type = ld_targ.t_m.m_sht_unwind;
1257         } else {
1258                 osp->os_shdr->sh_type = shdr->sh_type;
1259         }

1261         osp->os_shdr->sh_flags = shdr->sh_flags;
1262         osp->os_shdr->sh_entsize = shdr->sh_entsize;
1263         osp->os_name = oname;
1264         osp->os_namehash = onamehash;
1265         osp->os_ordndx = os_ndx;
1266         osp->os_sgdesc = sgp;
1267         if (is_ehframe)
1268                 osp->os_flags |= FLG_OS_EHFRAME;

1270         if (ifl && (shdr->sh_type == SHT_PROGBITS)) {
1271                 /*
1272                  * Try to preserve the intended meaning of sh_link/sh_info.
1273                  * See the translate_link() in update.c.
1274                  */
1275                 osp->os_shdr->sh_link = shdr->sh_link;
1276                 if (shdr->sh_flags & SHF_INFO_LINK)
1277                         osp->os_shdr->sh_info = shdr->sh_info;
1278         }

1280         /*
1281          * When -zignore is in effect, user supplied sections and files that are
1282          * not referenced from other sections, are eliminated from the object
1283          * being produced.  Some sections, although unreferenced, are special,
1284          * and must not be eliminated.  Determine if this new output section is
1285          * one of those special sections, and if so mark it artificially as
1286          * referenced.  Any input section and file associated to this output
1287          * section is also be marked as referenced, and thus won't be eliminated
1288          * from the final output.
1289          */
1290         if (ifl && ((ofl->ofl_flags1 & FLG_OF1_IGNPRC) || DBG_ENABLED)) {
1291                 const Msg       *refsec;

1293                 for (refsec = RefSecs; *refsec; refsec++) {
1294                         if (strcmp(osp->os_name, MSG_ORIG(*refsec)) == 0) {
1295                                 osp->os_flags |= FLG_OS_SECTREF;

1297                                 if ((ifl->ifl_flags & FLG_IF_IGNORE) ||
1298                                     DBG_ENABLED) {
1299                                         isp->is_flags |= FLG_IS_SECTREF;
1300                                         ifl->ifl_flags |= FLG_IF_FILEREF;
1301                                 }
```



```
1302                                 break;
1303                         }
1304                 }
1305         }

1307         /*
1308          * Sections of type SHT_GROUP are added to the ofl->ofl_osgroups list,
1309          * so that they can be updated as a group later.
1310          */
1311         if ((shdr->sh_type == SHT_GROUP) &&
1312             ((isp->is_flags & FLG_IS_DISCARD) == 0) &&
1313             (aplist_append(&ofl->ofl_osgroups, osp,
1314             AL_CNT_OFL_OSGROUPS) == NULL))
1315                 return ((Os_desc *)S_ERROR);

1317         /*
1318          * If this section is a non-empty TLS section indicate that a PT_TLS
1319          * program header is required.
1320          */
1321         if ((shflags & SHF_TLS) && shdr->sh_size &&
1322             ((ofl->ofl_flags & FLG_OF_RELOBJ) == 0))
1323                 ofl->ofl_flags |= FLG_OF_TLSPHDR;

1325         /*
1326          * If a non-allocatable section is going to be put into a loadable
1327          * segment then turn on the allocate bit for this section and warn the
1328          * user that we have done so.  This could only happen through the use
1329          * of a mapfile.
1330          */
1331         if ((sgp->sg_phdr.p_type == PT_LOAD) &&
1332             ((osp->os_shdr->sh_flags & SHF_ALLOC) == 0)) {
1333                 ld_eprintf(ofl, ERR_WARNING, MSG_INTL(MSG_SCN_NONALLOC),
1334                     ofl->ofl_name, osp->os_name, sgp->sg_name);
1335                 osp->os_shdr->sh_flags |= SHF_ALLOC;
1336         }

1338         /*
1339          * Retain this sections identifier for future comparisons when placing
1340          * a section (after all sections have been processed this variable will
1341          * be used to hold the sections symbol index as we don't need to retain
1342          * the identifier any more).
1343          */
1344         osp->os_identndx = ident;

1346         /*
1347          * Set alignment.
1348          */
1349         set_addralign(ofl, osp, isp);

1351         if (os_attach_isp(ofl, osp, isp, 0) == 0)
1352                 return ((Os_desc *)S_ERROR);

1354         DBG_CALL(Dbg_sec_created(ofl->ofl_lml, osp, sgp));

1356         /*
1357          * Insert the new section at the offset given by iidx.  If no position
1358          * for it was identified above, this will be index 0, causing the new
1359          * section to be prepended to the beginning of the section list.
1360          * Otherwise, it is the index following the section that was identified.
1361          */
1362         if (aplist_insert(&sgp->sg_osdescs, osp, AL_CNT_SG_OSDESC,
1363             iidx) == NULL)
1364                 return ((Os_desc *)S_ERROR);
1365         return (osp);
1366 }
```

```
**********************************************************
   88524 Thu Aug 25 11:51:59 2016
new/usr/src/cmd/sgs/packages/common/SUNWonld-README
7323 ld(1) -zignore can erroneously discard init and fini arrays as unreferenced
**********************************************************
```

```
#
# 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 #endif /* ! codereview */
```

```
**********************************************************
    3879 Thu Aug 25 11:52:00 2016
new/usr/src/cmd/sgs/rtld/i386/boot.s
7324 stack needs to be sufficiently aligned for SSE before init_array are called
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License, Version 1.0 only
 * (the "License").  You may not use this file except in compliance
 * with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 *      Copyright (c) 1988 AT&T
 *        All Rights Reserved
 *
 *
 * Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */
#pragma ident   "%Z%%M% %I%     %E% SMI"

/*
 * Bootstrap routine for run-time linker.
 * We get control from exec which has loaded our text and
 * data into the process' address space and created the process
 * stack.
 *
 * On entry, the process stack looks like this:
 *
 *      #                       # <- %esp
 *      #_______________________#  high addresses
 *      #       strings         #
 *      #_______________________#
 *      #       0 word          #
 *      #_______________________#
 *      #       Auxiliary       #
 *      #       entries         #
 *      #       ...             #
 *      #       (size varies)   #
 *      #_______________________#
 *      #       0 word          #
 *      #_______________________#
 *      #       Environment     #
 *      #       pointers        #
 *      #       ...             #
 *      #       (one word each) #
 *      #_______________________#
 *      #       0 word          #
 *      #_______________________#
 *      #       Argument        # low addresses
 *      #       pointers        #
```



```
 *      #       Argc words      #
 *      #_______________________#
 *      #       argc            #
 *      #_______________________# <- %ebp
 *
 *
 * We must calculate the address at which ld.so was loaded,
 * find the addr of the dynamic section of ld.so, of argv[0], and  of
 * the process' environment pointers - and pass the thing to _setup
 * to handle.  We then call _rtld - on return we jump to the entry
 * point for the a.out.
 */

#if     defined(lint)

extern  unsigned long   _setup();
extern  void            atexit_fini();
void
main()
{
        (void) _setup();
        atexit_fini();
}

#else

#include        <link.h>

        .file   "boot.s"
        .text
        .globl  _rt_boot
        .globl  _setup
        .globl  _GLOBAL_OFFSET_TABLE_
        .type   _rt_boot,@function
        .align  4

        / init is called from the _init symbol in the CRT, however .init_array
        / are called "naturally" from call_init.  Because of that, we need the
        / stack aligned here so that initializers called via _array sections may
        / safely use SIMD instructions.
#endif /* ! codereview */
_rt_alias:
        jmp     .get_ip                 / in case we were invoked from libc.so
_rt_boot:
        movl    %esp,%ebp               / save for referencing args
        subl    $EB_MAX_SIZE32,%esp     / make room for a max sized boot vector
        andl    $-16,%esp
        subl    $8,%esp
#endif /* ! codereview */
        movl    %esp,%esi               / use esi as a pointer to &eb[0]
        movl    $EB_ARGV,0(%esi)        / set up tag for argv
        leal    4(%ebp),%eax            / get address of argv
        movl    %eax,4(%esi)            / put after tag
        movl    $EB_ENVP,8(%esi)        / set up tag for envp
        movl    (%ebp),%eax             / get # of args
        addl    $2,%eax                 / one for the zero & one for argc
        leal    (%ebp,%eax,4),%edi      / now points past args & @ envp
        movl    %edi,12(%esi)           / set envp
.L0:    addl    $4,%edi                 / next
        cmpl    $0,-4(%edi)             / search for 0 at end of env
        jne     .L0
        movl    $EB_AUXV,16(%esi)       / set up tag for auxv
        movl    %edi,20(%esi)           / point to auxv
        movl    $EB_NULL,24(%esi)       / set up NULL tag
.get_ip:
        call    .L1                     / only way to get IP into a register
```

```
.L1:
        popl    %ebx                    / pop the IP we just "pushed"
        addl    $_GLOBAL_OFFSET_TABLE_+[.-.L1],%ebx
        pushl   (%ebx)                  / address of dynamic structure
        pushl   %esi                    / push &eb[0]

        call    _setup@PLT              / _setup(&eb[0], _DYNAMIC)
        movl    %ebp,%esp               / release stack frame

        movl    atexit_fini@GOT(%ebx), %edx
        jmp     *%eax                   / transfer control to a.out
        .size   _rt_boot,.-_rt_boot

#endif
```

```
**********************************************************
   59048 Thu Aug 25 11:52:01 2016
new/usr/src/man/man1/ld.1
7322 ld(1) gets init/fini array section names wrong
**********************************************************
```

```
   1 '\" te
   2 .\" Copyright 1989 AT&T
   3 .\" Copyright (c) 2009, Sun Microsystems, Inc. All Rights Reserved
   4 .\" Copyright (c) 2012, Joyent, Inc. All Rights Reserved
   5 .\" The contents of this file are subject to the terms of the Common Development
   6 .\"  See the License for the specific language governing permissions and limitat
   7 .\" the fields enclosed by brackets "[]" replaced with your own identifying info
   8 .TH LD 1 "Sep 10, 2013"
   9 .SH NAME
  10 ld \- link-editor for object files
  11 .SH SYNOPSIS
  12 .LP
  13 .nf
  14 \fBld\fR [\fB-32\fR | \fB-64\fR] [\fB-a\fR | \fB-r\fR] [\fB-b\fR] [\fB-B\fRdirec
  15 [\fB-B\fR dynamic | static] [\fB-B\fR eliminate] [\fB-B\fR group] [\fB-B\fR loca
  16 [\fB-B\fR reduce] [\fB-B\fR symbolic] [\fB-c\fR \fIname\fR] [\fB-C\fR] [\fB-d\fR
  17 [\fB-D\fR \fItoken\fR,...] [\fB-e\fR \fIepsym\fR] [\fB-f\fR \fIname\fR | \fB-F\f
  18 [\fB-i\fR] [\fB-I\fR \fIname\fR] [\fB-l\fR \fIx\fR] [\fB-L\fR \fIpath\fR] [\fB-m
  19 [\fB-N\fR \fIstring\fR] [\fB-o\fR \fIoutfile\fR] [\fB-p\fR \fIauditlib\fR] [\fB-
  20 [\fB-Q\fR y | n] [\fB-R\fR \fIpath\fR] [\fB-s\fR] [\fB-S\fR \fIsupportlib\fR] [\
  21 [\fB-u\fR \fIsymname\fR] [\fB-V\fR] [\fB-Y P\fR\fI,dirlist\fR] [\fB-z\fR absexec
  22 [\fB-z\fR allextract | defaultextract | weakextract ] [\fB-z\fR altexec64]
  23 [\fB-z\fR assert-deflib ] [ \fB-z\fR assert-deflib=\fIlibname\fR ]
  24 [\fB-z\fR combreloc | nocombreloc ] [\fB-z\fR defs | nodefs]
  25 [\fB-z\fR direct | nodirect] [\fB-z\fR endfiltee]
  26 [\fB-z\fR fatal-warnings | nofatal-warnings ] [\fB-z\fR finiarray=\fIfunction\fR
  27 [\fB-z\fR globalaudit] [\fB-z\fR groupperm | nogroupperm]
  28 [\fB-z\fR guidance[=\fIid1\fR,\fIid2\fR...] [\fB-z\fR help ]
  29 [\fB-z\fR ignore | record] [\fB-z\fR initarray=\fIfunction\fR] [\fB-z\fR initfir
  30 [\fB-z\fR interpose] [\fB-z\fR lazyload | nolazyload]
  31 [\fB-z\fR ld32=\fIarg1\fR,\fIarg2\fR,...] [\fB-z\fR ld64=\fIarg1\fR,\fIarg2\fR,.
  32 [\fB-z\fR loadfltr] [\fB-z\fR muldefs] [\fB-z\fR nocompstrtab] [\fB-z\fR nodefau
  33 [\fB-z\fR nodelete] [\fB-z\fR nodlopen] [\fB-z\fR nodump] [\fB-z\fR noldynsym]
  34 [\fB-z\fR nopartial] [\fB-z\fR noversion] [\fB-z\fR now] [\fB-z\fR origin]
  35 [\fB-z\fR preinitarray=\fIfunction\fR] [\fB-z\fR redlocsym] [\fB-z\fR relaxreloc
  36 [\fB-z\fR rescan-now] [\fB-z\fR recan] [\fB-z\fR rescan-start \fI\&...\fR \fB-z\
  37 [\fB-z\fR target=sparc|x86] [\fB-z\fR text | textwarn | textoff]
  38 [\fB-z\fR verbose] [\fB-z\fR wrap=\fIsymbol\fR] \fIfilename\fR...
  39 .fi

  41 .SH DESCRIPTION
  42 .sp
  43 .LP
  44 The link-editor, \fBld\fR, combines relocatable object files by resolving
  45 symbol references to symbol definitions, together with performing relocations.
  46 \fBld\fR operates in two modes, static or dynamic, as governed by the \fB-d\fR
  47 option. In all cases, the output of \fBld\fR is left in the file \fBa.out\fR by
  48 default. See NOTES.
  49 .sp
  50 .LP
  51 In dynamic mode, \fB-dy\fR, the default, relocatable object files that are
  52 provided as arguments are combined to produce an executable object file. This
  53 file is linked at execution with any shared object files that are provided as
  54 arguments. If the \fB-G\fR option is specified, relocatable object files are
  55 combined to produce a shared object. Without the \fB-G\fR option, a dynamic
  56 executable is created.
  57 .sp
  58 .LP
  59 In static mode, \fB-dn\fR, relocatable object files that are provided as
  60 arguments are combined to produce a static executable file. If the \fB-r\fR
  61 option is specified, relocatable object files are combined to produce one
```



```
  62 relocatable object file. See \fBStatic Executables\fR.
  63 .sp
  64 .LP
  65 Dynamic linking is the most common model for combining relocatable objects, and
  66 the eventual creation of processes within Solaris. This environment tightly
  67 couples the work of the link-editor and the runtime linker, \fBld.so.1\fR(1).
  68 Both of these utilities, together with their related technologies and
  69 utilities, are extensively documented in the \fILinker and Libraries Guide\fR.
  70 .sp
  71 .LP
  72 If any argument is a library, \fBld\fR by default searches the library exactly
  73 once at the point the library is encountered on the argument list. The library
  74 can be either a shared object or relocatable archive. See \fBar.h\fR(3HEAD)).
  75 .sp
  76 .LP
  77 A shared object consists of an indivisible, whole unit that has been generated
  78 by a previous link-edit of one or more input files. When the link-editor
  79 processes a shared object, the entire contents of the shared object become a
  80 logical part of the resulting output file image. The shared object is not
  81 physically copied during the link-edit as its actual inclusion is deferred
  82 until process execution. This logical inclusion means that all symbol entries
  83 defined in the shared object are made available to the link-editing process.
  84 See Chapter 4, \fIShared Objects,\fR in \fILinker and Libraries Guide\fR
  85 .sp
  86 .LP
  87 For an archive library, \fBld\fR loads only those routines that define an
  88 unresolved external reference. \fBld\fR searches the symbol table of the
  89 archive library sequentially to resolve external references that can be
  90 satisfied by library members. This search is repeated until no external
  91 references can be resolved by the archive. Thus, the order of members in the
  92 library is functionally unimportant, unless multiple library members exist that
  93 define the same external symbol. Archive libraries that have interdependencies
  94 can require multiple command line definitions, or the use of one of the
  95 \fB-z\fR \fBrescan\fR options. See \fIArchive Processing\fR in \fILinker and
  96 Libraries Guide\fR.
  97 .sp
  98 .LP
  99 \fBld\fR is a cross link-editor, able to link 32-bit objects or 64-bit objects,
 100 for Sparc or x86 targets. \fBld\fR uses the \fBELF\fR class and machine type of
 101 the first relocatable object on the command line to govern the mode in which to
 102 operate. The mixing of 32-bit objects and 64-bit objects is not permitted.
 103 Similarly, only objects of a single machine type are allowed. See the
 104 \fB-32\fR, \fB-64\fR and \fB-z target\fR options, and the \fBLD_NOEXEC_64\fR
 105 environment variable.
 106 .SS "Static Executables"
 107 .sp
 108 .LP
 109 The creation of static executables has been discouraged for many releases. In
 110 fact, 64-bit system archive libraries have never been provided. Because a
 111 static executable is built against system archive libraries, the executable
 112 contains system implementation details. This self-containment has a number of
 113 drawbacks.
 114 .RS +4
 115 .TP
 116 .ie t \(bu
 117 .el o
 118 The executable is immune to the benefits of system patches delivered as shared
 119 objects. The executable therefore, must be rebuilt to take advantage of many
 120 system improvements.
 121 .RE
 122 .RS +4
 123 .TP
 124 .ie t \(bu
 125 .el o
 126 The ability of the executable to run on future releases can be compromised.
 127 .RE
```

```
.RS +4
.TP
.ie t \(bu
.el o
The duplication of system implementation details negatively affects system
performance.
.RE
.sp
.LP
With Solaris 10, 32-bit system archive libraries are no longer provided.
Without these libraries, specifically \fBlibc.a\fR, the creation of static
executables is no longer achievable without specialized system knowledge.
However, the capability of \fBld\fR to process static linking options, and the
processing of archive libraries, remains unchanged.
.SH OPTIONS
.sp
.LP
The following options are supported.
.sp
.ne 2
.na
\fB\fB-32\fR | \fB-64\fR\fR
.ad
.sp .6
.RS 4n
Creates a 32-bit, or 64-bit object.
.sp
By default, the class of the object being generated is determined from the
first \fBELF\fR object processed from the command line. If no objects are
specified, the class is determined by the first object encountered within the
first archive processed from the command line. If there are no objects or
archives, the link-editor creates a 32-bit object.
.sp
The \fB-64\fR option is required to create a 64-bit object solely from a
mapfile.
.sp
This \fB-32\fR or \fB-64\fR options can also be used in the rare case of
linking entirely from an archive that contains a mixture of 32 and 64-bit
objects. If the first object in the archive is not the class of the object that
is required to be created, then the \fB-32\fR or \fB-64\fR option can be used
to direct the link-editor. See \fIThe 32-bit link-editor and 64-bit
link-editor\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-a\fR\fR
.ad
.sp .6
.RS 4n
In static mode only, produces an executable object file. Undefined references
are not permitted. This option is the default behavior for static mode. The
\fB-a\fR option can not be used with the \fB-r\fR option. See \fBStatic
Executables\fR under DESCRIPTION.
.RE

.sp
.ne 2
.na
\fB\fB-b\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, provides no special processing for dynamic executable
relocations that reference symbols in shared objects. Without the \fB-b\fR
```



```
option, the link-editor applies techniques within a dynamic executable so that
the text segment can remain read-only. One technique is the creation of special
position-independent relocations for references to functions that are defined
in shared objects. Another technique arranges for data objects that are defined
in shared objects to be copied into the memory image of an executable at
runtime.
.sp
The \fB-b\fR option is intended for specialized dynamic objects and is not
recommended for general use. Its use suppresses all specialized processing
required to ensure an object's shareability, and can even prevent the
relocation of 64-bit executables.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBdirect\fR | \fBnodirect\fR\fR
.ad
.sp .6
.RS 4n
These options govern direct binding. \fB-B\fR \fBdirect\fR establishes direct
binding information by recording the relationship between each symbol reference
together with the dependency that provides the definition. In addition, direct
binding information is established between each symbol reference and an
associated definition within the object being created. The runtime linker uses
this information to search directly for a symbol in the associated object
rather than to carry out a default symbol search.
.sp
Direct binding information can only be established to dependencies specified
with the link-edit. Thus, you should use the \fB-z\fR \fBdefs\fR option.
Objects that wish to interpose on symbols in a direct binding environment
should identify themselves as interposers with the \fB-z\fR \fBinterpose\fR
option. The use of \fB-B\fR \fBdirect\fR enables \fB-z\fR \fBlazyload\fR for
all dependencies.
.sp
The \fB-B\fR \fBnodirect\fR option prevents any direct binding to the
interfaces offered by the object being created. The object being created can
continue to directly bind to external interfaces by specifying the \fB-z\fR
\fBdirect\fR option. See Appendix D, \fIDirect Bindings,\fR in \fILinker and
Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBdynamic\fR | \fBstatic\fR\fR
.ad
.sp .6
.RS 4n
Options governing library inclusion. \fB-B\fR \fBdynamic\fR is valid in dynamic
mode only. These options can be specified any number of times on the command
line as toggles: if the \fB-B\fR \fBstatic\fR option is given, no shared
objects are accepted until \fB-B\fR \fBdynamic\fR is seen. See the \fB-l\fR
option.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBeliminate\fR\fR
.ad
.sp .6
.RS 4n
Causes any global symbols, not assigned to a version definition, to be
eliminated from the symbol table. Version definitions can be supplied by means
of a \fBmapfile\fR to indicate the global symbols that should remain visible in
```

the generated object. This option achieves the same symbol elimination as the \fIauto-elimination\fR directive that is available as part of a \fBmapfile\fR version definition. This option can be useful when combining versioned and non-versioned relocatable objects. See also the \fB-B\fR \fBlocal\fR option and the \fB-B\fR \fBreduce\fR option. See \fIDefining Additional Symbols with a mapfile\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBgroup\fR\fR
.ad
.sp .6
.RS 4n
Establishes a shared object and its dependencies as a group. Objects within the group are bound to other members of the group at runtime. This mode is similar to adding the object to the process by using \fBdlopen\fR(3C) with the \fBRTLD_GROUP\fR mode. An object that has an explicit dependency on a object identified as a group, becomes a member of the group.
.sp
As the group must be self contained, use of the \fB-B\fR \fBgroup\fR option also asserts the \fB-z\fR \fBdefs\fR option.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBlocal\fR\fR
.ad
.sp .6
.RS 4n
Causes any global symbols, not assigned to a version definition, to be reduced to local. Version definitions can be supplied by means of a \fBmapfile\fR to indicate the global symbols that should remain visible in the generated object. This option achieves the same symbol reduction as the \fIauto-reduction\fR directive that is available as part of a \fBmapfile\fR version definition. This option can be useful when combining versioned and non-versioned relocatable objects. See also the \fB-B\fR \fBeliminate\fR option and the \fB-B\fR \fBreduce\fR option. See \fIDefining Additional Symbols with a mapfile\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBreduce\fR\fR
.ad
.sp .6
.RS 4n
When generating a relocatable object, causes the reduction of symbolic information defined by any version definitions. Version definitions can be supplied by means of a \fBmapfile\fR to indicate the global symbols that should remain visible in the generated object. By default, when a relocatable object is generated, version definitions are only recorded in the output image. The actual reduction of symbolic information is carried out when the object is used in the construction of a dynamic executable or shared object. The \fB-B\fR \fBreduce\fR option is applied automatically when a dynamic executable or shared object is created.
.RE

.sp
.ne 2
.na
\fB\fB-B\fR \fBsymbolic\fR\fR
.ad

.sp .6
.RS 4n
In dynamic mode only. When building a shared object, binds references to global symbols to their definitions, if available, within the object. Normally, references to global symbols within shared objects are not bound until runtime, even if definitions are available. This model allows definitions of the same symbol in an executable or other shared object to override the object's own definition. \fBld\fR issues warnings for undefined symbols unless \fB-z\fR \fBdefs\fR overrides.
.sp
The \fB-B\fR \fBsymbolic\fR option is intended for specialized dynamic objects and is not recommended for general use. To reduce the runtime relocation processing that is required an object, the creation of a version definition is recommended.
.RE

.sp
.ne 2
.na
\fB\fB-c\fR \fIname\fR\fR
.ad
.sp .6
.RS 4n
Records the configuration file \fIname\fR for use at runtime. Configuration files can be employed to alter default search paths, provide a directory cache, together with providing alternative object dependencies. See \fBcrle\fR(1).
.RE

.sp
.ne 2
.na
\fB\fB-C\fR\fR
.ad
.sp .6
.RS 4n
Demangles C++ symbol names displayed in diagnostic messages.
.RE

.sp
.ne 2
.na
\fB\fB-d\fR \fBy\fR | \fBn\fR\fR
.ad
.sp .6
.RS 4n
When \fB-d\fR \fBy\fR, the default, is specified, \fBld\fR uses dynamic linking. When \fB-d\fR \fBn\fR is specified, \fBld\fR uses static linking. See \fBStatic Executables\fR under DESCRIPTION, and \fB-B\fR \fBdynamic\fR|\fBstatic\fR.
.RE

.sp
.ne 2
.na
\fB\fB-D\fR \fItoken\fR,...\fR
.ad
.sp .6
.RS 4n
Prints debugging information as specified by each \fItoken\fR, to the standard error. The special token \fBhelp\fR indicates the full list of tokens available. See \fIDebugging Aids\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na

```
\fB\fB-e\fR \fIepsym\fR\fR
.ad
.br
.na
\fB\fB--entry\fR \fIepsym\fR\fR
.ad
.sp .6
.RS 4n
Sets the entry point address for the output file to be the symbol \fIepsym\fR.
.RE

.sp
.ne 2
.na
\fB\fB-f\fR \fIname\fR\fR
.ad
.br
.na
\fB\fB--auxiliary\fR \fIname\fR\fR
.ad
.sp .6
.RS 4n
Useful only when building a shared object. Specifies that the symbol table of
the shared object is used as an auxiliary filter on the symbol table of the
shared object specified by \fIname\fR. Multiple instances of this option are
allowed. This option can not be combined with the \fB-F\fR option. See
\fIGenerating Auxiliary Filters\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-F\fR \fIname\fR\fR
.ad
.br
.na
\fB\fB--filter\fR \fIname\fR\fR
.ad
.sp .6
.RS 4n
Useful only when building a shared object. Specifies that the symbol table of
the shared object is used as a filter on the symbol table of the shared object
specified by \fIname\fR. Multiple instances of this option are allowed. This
option can not be combined with the \fB-f\fR option. See \fIGenerating Standard
Filters\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-G\fR\fR
.ad
.br
.na
\fB\fB-shared\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, produces a shared object. Undefined symbols are allowed.
See Chapter 4, \fIShared Objects,\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-h\fR \fIname\fR\fR
```



```
.ad
.br
.na
\fB\fB--soname\fR \fIname\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, when building a shared object, records \fIname\fR in the
object's dynamic section. \fIname\fR is recorded in any dynamic objects that
are linked with this object rather than the object's file system name.
Accordingly, \fIname\fR is used by the runtime linker as the name of the shared
object to search for at runtime. See \fIRecording a Shared Object Name\fR in
\fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-i\fR\fR
.ad
.sp .6
.RS 4n
Ignores \fBLD_LIBRARY_PATH\fR. This option is useful when an
\fBLD_LIBRARY_PATH\fR setting is in effect to influence the runtime library
search, which would interfere with the link-editing being performed.
.RE

.sp
.ne 2
.na
\fB\fB-I\fR \fIname\fR\fR
.ad
.br
.na
\fB\fB--dynamic-linker\fR \fIname\fR\fR
.ad
.sp .6
.RS 4n
When building an executable, uses \fIname\fR as the path name of the
interpreter to be written into the program header. The default in static mode
is no interpreter. In dynamic mode, the default is the name of the runtime
linker, \fBld.so.1\fR(1). Either case can be overridden by \fB-I\fR \fIname\fR.
\fBexec\fR(2) loads this interpreter when the \fBa.out\fR is loaded, and passes
control to the interpreter rather than to the \fBa.out\fR directly.
.RE

.sp
.ne 2
.na
\fB\fB-l\fR \fIx\fR\fR
.ad
.br
.na
\fB\fB--library\fR \fIx\fR\fR
.ad
.sp .6
.RS 4n
Searches a library \fBlib\fR\fIx\fR\fB\&.so\fR or \fBlib\fR\fIx\fR\fB\&.a\fR,
the conventional names for shared object and archive libraries, respectively.
In dynamic mode, unless the \fB-B\fR \fBstatic\fR option is in effect, \fBld\fR
searches each directory specified in the library search path for a
\fBlib\fR\fIx\fR\fB\&.so\fR or \fBlib\fR\fIx\fR\fB\&.a\fR file. The directory
search stops at the first directory containing either. \fBld\fR chooses the
file ending in \fB\&.so\fR if \fB-l\fR\fIx\fR expands to two files with names
of the form \fBlib\fR\fIx\fR\fB\&.so\fR and \fBlib\fR\fIx\fR\fB\&.a\fR. If no
\fBlib\fR\fIx\fR\fB\&.so\fR is found, then \fBld\fR accepts
```

\fBlib\fR\fIx\fR\fB\&.a\fR. In static mode, or when the \fB-B\fR \fBstatic\fR option is in effect, \fBld\fR selects only the file ending in \fB\&.a\fR. \fBld\fR searches a library when the library is encountered, so the placement of \fB-l\fR is significant. See \fILinking With Additional Libraries\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-L\fR \fIpath\fR\fR
.ad
.br
.na
\fB\fB--library-path\fR \fIpath\fR\fR
.ad
.sp .6
.RS 4n
Adds \fIpath\fR to the library search directories. \fBld\fR searches for libraries first in any directories specified by the \fB-L\fR options and then in the standard directories. This option is useful only if the option precedes the \fB-l\fR options to which the \fB-L\fR option applies. See \fIDirectories Searched by the Link-Editor\fR in \fILinker and Libraries Guide\fR.
.sp
The environment variable \fBLD_LIBRARY_PATH\fR can be used to supplement the library search path, however the \fB-L\fR option is recommended, as the environment variable is also interpreted by the runtime environment. See \fBLD_LIBRARY_PATH\fR under ENVIRONMENT VARIABLES.
.RE

.sp
.ne 2
.na
\fB\fB-m\fR\fR
.ad
.sp .6
.RS 4n
Produces a memory map or listing of the input/output sections, together with any non-fatal multiply-defined symbols, on the standard output.
.RE

.sp
.ne 2
.na
\fB\fB-M\fR \fImapfile\fR\fR
.ad
.sp .6
.RS 4n
Reads \fImapfile\fR as a text file of directives to \fBld\fR. This option can be specified multiple times. If \fImapfile\fR is a directory, then all regular files, as defined by \fBstat\fR(2), within the directory are processed. See Chapter 9, \fIMapfile Option,\fR in \fILinker and Libraries Guide\fR. Example mapfiles are provided in \fB/usr/lib/ld\fR. See FILES.
.RE

.sp
.ne 2
.na
\fB\fB-N\fR \fIstring\fR\fR
.ad
.sp .6
.RS 4n
This option causes a \fBDT_NEEDED\fR entry to be added to the \fB\&.dynamic\fR section of the object being built. The value of the \fBDT_NEEDED\fR string is the \fIstring\fR that is specified on the command line. This option is position dependent, and the \fBDT_NEEDED\fR \fB\&.dynamic\fR entry is relative to the

other dynamic dependencies discovered on the link-edit line. This option is useful for specifying dependencies within device driver relocatable objects when combined with the \fB-dy\fR and \fB-r\fR options.
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIoutfile\fR\fR
.ad
.br
.na
\fB\fB--output\fR \fIoutfile\fR\fR
.ad
.sp .6
.RS 4n
Produces an output object file that is named \fIoutfile\fR. The name of the default object file is \fBa.out\fR.
.RE

.sp
.ne 2
.na
\fB\fB-p\fR \fIauditlib\fR\fR
.ad
.sp .6
.RS 4n
Identifies an audit library, \fIauditlib\fR. This audit library is used to audit the object being created at runtime. A shared object identified as requiring auditing with the \fB-p\fR option, has this requirement inherited by any object that specifies the shared object as a dependency. See the \fB-P\fR option. See \fIRuntime Linker Auditing Interface\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-P\fR \fIauditlib\fR\fR
.ad
.sp .6
.RS 4n
Identifies an audit library, \fIauditlib\fR. This audit library is used to audit the dependencies of the object being created at runtime. Dependency auditing can also be inherited from dependencies that are identified as requiring auditing. See the \fB-p\fR option, and the \fB-z\fR \fBglobalaudit\fR option. See \fIRuntime Linker Auditing Interface\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-Q\fR \fBy\fR | \fBn\fR\fR
.ad
.sp .6
.RS 4n
Under \fB-Q\fR \fBy\fR, an \fBident\fR string is added to the \fB\&.comment\fR section of the output file. This string identifies the version of the \fBld\fR used to create the file. This results in multiple \fBld\fR \fBidents\fR when there have been multiple linking steps, such as when using \fBld\fR \fB-r\fR. This identification is identical with the default action of the \fBcc\fR command. \fB-Q\fR \fBn\fR suppresses version identification. \fB\&.comment\fR sections can be manipulated by the \fBmcs\fR(1) utility.
.RE

```
.sp
.ne 2
.na
\fB\fB-r\fR\fR
.ad
.br
.na
\fB\fB--relocatable\fR\fR
.ad
.sp .6
.RS 4n
Combines relocatable object files to produce one relocatable object file.
\fBld\fR does not complain about unresolved references. This option cannot be
used with the \fB-a\fR option.
.RE

.sp
.ne 2
.na
\fB\fB-R\fR \fIpath\fR\fR
.ad
.br
.na
\fB\fB-rpath\fR \fIpath\fR\fR
.ad
.sp .6
.RS 4n
A colon-separated list of directories used to specify library search
directories to the runtime linker. If present and not NULL, the path is
recorded in the output object file and passed to the runtime linker. Multiple
instances of this option are concatenated together with each \fIpath\fR
separated by a colon. See \fIDirectories Searched by the Runtime Linker\fR in
\fILinker and Libraries Guide\fR.
.sp
The use of a runpath within an associated object is preferable to setting
global search paths such as through the \fBLD_LIBRARY_PATH\fR environment
variable. Only the runpaths that are necessary to find the objects dependencies
should be recorded. \fBldd\fR(1) can also be used to discover unused runpaths
in dynamic objects, when used with the \fB-U\fR option.
.sp
Various tokens can also be supplied with a runpath that provide a flexible
means of identifying system capabilities or an objects location. See Appendix
C, \fIEstablishing Dependencies with Dynamic String Tokens,\fR in \fILinker and
Libraries Guide\fR. The \fB$ORIGIN\fR token is especially useful in allowing
dynamic objects to be relocated to different locations in the file system.
.RE

.sp
.ne 2
.na
\fB\fB-s\fR\fR
.ad
.br
.na
\fB\fB--strip-all\fR\fR
.ad
.sp .6
.RS 4n
Strips symbolic information from the output file. Any debugging information,
that is, \fB\&.line\fR, \fB\&.debug*\fR, and \fB\&.stab*\fR sections, and their
associated relocation entries are removed. Except for relocatable files, a
symbol table \fBSHT_SYMTAB\fR and its associated string table section are not
created in the output object file. The elimination of a \fBSHT_SYMTAB\fR symbol
table can reduce the \fB\&.stab*\fR debugging information that is generated
using the compiler drivers \fB-g\fR option. See the \fB-z\fR \fBredlocsym\fR
and \fB-z\fR \fBnoldynsym\fR options.
```



```
.RE

.sp
.ne 2
.na
\fB\fB-S\fR \fIsupportlib\fR\fR
.ad
.sp .6
.RS 4n
The shared object \fIsupportlib\fR is loaded with \fBld\fR and given
information regarding the linking process. Shared objects that are defined by
using the \fB-S\fR option can also be supplied using the \fBSGS_SUPPORT\fR
environment variable. See \fILink-Editor Support Interface\fR in \fILinker and
Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-t\fR\fR
.ad
.sp .6
.RS 4n
Turns off the warning for multiply-defined symbols that have different sizes or
different alignments.
.RE

.sp
.ne 2
.na
\fB\fB-u\fR \fIsymname\fR\fR
.ad
.br
.na
\fB\fB--undefined\fR \fIsymname\fR\fR
.ad
.sp .6
.RS 4n
Enters \fIsymname\fR as an undefined symbol in the symbol table. This option is
useful for loading entirely from an archive library. In this instance, an
unresolved reference is needed to force the loading of the first routine. The
placement of this option on the command line is significant. This option must
be placed before the library that defines the symbol. See \fIDefining
Additional Symbols with the u option\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-V\fR\fR
.ad
.br
.na
\fB\fB--version\fR\fR
.ad
.sp .6
.RS 4n
Outputs a message giving information about the version of \fBld\fR being used.
.RE

.sp
.ne 2
.na
\fB\fB-Y\fR \fBP,\fR\fIdirlist\fR\fR
.ad
.sp .6
```

```
.RS 4n
Changes the default directories used for finding libraries. \fIdirlist\fR is a
colon-separated path list.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBabsexec\fR\fR
.ad
.sp .6
.RS 4n
Useful only when building a dynamic executable. Specifies that references to
external absolute symbols should be resolved immediately instead of being left
for resolution at runtime. In very specialized circumstances, this option
removes text relocations that can result in excessive swap space demands by an
executable.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBallextract\fR | \fBdefaultextract\fR | \fBweakextract\fR\fR
.ad
.br
.na
\fB\fB--whole-archive\fR | \fB--no-whole-archive\fR\fR
.ad
.sp .6
.RS 4n
Alters the extraction criteria of objects from any archives that follow. By
default, archive members are extracted to satisfy undefined references and to
promote tentative definitions with data definitions. Weak symbol references do
not trigger extraction. Under the \fB-z\fR \fBallextract\fR or
\fB--whole-archive\fR options, all archive members are extracted from the
archive. Under \fB-z\fR \fBweakextract\fR, weak references trigger archive
extraction. The \fB-z\fR \fBdefaultextract\fR or \fB--no-whole-archive\fR
options provide a means of returning to the default following use of the former
extract options. See \fIArchive Processing\fR in \fILinker and Libraries
Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBaltexec64\fR\fR
.ad
.sp .6
.RS 4n
Execute the 64-bit \fBld\fR. The creation of very large 32-bit objects can
exhaust the virtual memory that is available to the 32-bit \fBld\fR. The
\fB-z\fR \fBaltexec64\fR option can be used to force the use of the associated
64-bit \fBld\fR. The 64-bit \fBld\fR provides a larger virtual address space
for building 32-bit objects. See \fIThe 32-bit link-editor and 64-bit
link-editor\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBcombreloc\fR | \fBnocombreloc\fR\fR
.ad
.sp .6
.RS 4n
By default, \fBld\fR combines multiple relocation sections when building
executables or shared objects. This section combination differs from
```



```
relocatable objects, in which relocation sections are maintained in a
one-to-one relationship with the sections to which the relocations must be
applied. The \fB-z\fR \fBnocombreloc\fR option disables this merging of
relocation sections, and preserves the one-to-one relationship found in the
original relocatable objects.
.sp
\fBld\fR sorts the entries of data relocation sections by their symbol
reference. This sorting reduces runtime symbol lookup. When multiple relocation
sections are combined, this sorting produces the least possible relocation
overhead when objects are loaded into memory, and speeds the runtime loading of
dynamic objects.
.sp
Historically, the individual relocation sections were carried over to any
executable or shared object, and the \fB-z\fR \fBcombreloc\fR option was
required to enable the relocation section merging previously described.
Relocation section merging is now the default. The \fB-z\fR \fBcombreloc\fR
option is still accepted for the benefit of old build environments, but the
option is unnecessary, and has no effect.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBassert-deflib\fR\fR
.ad
.br
.na
\fB\fB-z\fR \fBassert-deflib=\fR\fIlibname\fR\fR
.ad
.sp .6
.RS 4n
Enables warnings that check the location of where libraries passed in with
\fB-l\fR are found. If the link-editor finds a library on its default search
path it will emit a warning. This warning can be made fatal in conjunction with
the option \fB-z fatal-warnings\fR. Passing \fIlibname\fR white lists a library
from this check. The library must be the full name of the library, e.g.
\fIlibc.so\fR. To white list multiple libraries, the \fB-z
assert-deflib=\fR\fIlibname\fR option can be repeated multiple times. This
option is useful when trying to build self-contained objects where a referenced
library might exist in the default system library path and in alternate paths
specified by \fB-L\fR, but you only want the alternate paths to be used.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBdefs\fR | \fBnodefs\fR\fR
.ad
.br
.na
\fB\fB--no-undefined\fR\fR
.ad
.sp .6
.RS 4n
The \fB-z\fR \fBdefs\fR option and the \fB--no-undefined\fR option force a
fatal error if any undefined symbols remain at the end of the link. This mode
is the default when an executable is built. For historic reasons, this mode is
\fBnot\fR the default when building a shared object. Use of the \fB-z\fR
\fBdefs\fR option is recommended, as this mode assures the object being built
is self-contained. A self-contained object has all symbolic references resolved
internally, or to the object's immediate dependencies.
.sp
The \fB-z\fR \fBnodefs\fR option allows undefined symbols. For historic
reasons, this mode is the default when a shared object is built. When used with
executables, the behavior of references to such undefined symbols is
unspecified. Use of the \fB-z\fR \fBnodefs\fR option is not recommended.
```

```
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBdirect\fR | \fBnodirect\fR\fR
.ad
.sp .6
.RS 4n
Enables or disables direct binding to any dependencies that follow on the
command line. These options allow finer control over direct binding than the
global counterpart \fB-B\fR \fBdirect\fR. The \fB-z\fR \fBdirect\fR option also
differs from the \fB-B\fR \fBdirect\fR option in the following areas. Direct
binding information is not established between a symbol reference and an
associated definition within the object being created. Lazy loading is not
enabled.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBendfiltee\fR\fR
.ad
.sp .6
.RS 4n
Marks a filtee so that when processed by a filter, the filtee terminates any
further filtee searches by the filter. See \fIReducing Filtee Searches\fR in
\fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBfatal-warnings\fR | \fBnofatal-warnings\fR\fR
.ad
.br
.na
\fB\fB--fatal-warnings\fR | \fB--no-fatal-warnings\fR
.ad
.sp .6
.RS 4n
Controls the behavior of warnings emitted from the link-editor. Setting \fB-z
fatal-warnings\fR promotes warnings emitted by the link-editor to fatal errors
that will cause the link-editor to fail before linking. \fB-z
nofatal-warnings\fR instead demotes these warnings such that they will not cause
the link-editor to exit prematurely.
.RE


.sp
.ne 2
.na
\fB\fB-z\fR \fBfiniarray=\fR\fIfunction\fR\fR
.ad
.sp .6
.RS 4n
Appends an entry to the \fB\&.fini_array\fR section of the object being built.
If no \fB\&.fini_array\fR section is present, a section is created. The new
Appends an entry to the \fB\&.finiarray\fR section of the object being built.
If no \fB\&.finiarray\fR section is present, a section is created. The new
entry is initialized to point to \fIfunction\fR. See \fIInitialization and
Termination Sections\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
```



```
.na
\fB\fB-z\fR \fBglobalaudit\fR\fR
.ad
.sp .6
.RS 4n
This option supplements an audit library definition that has been recorded with
the \fB-P\fR option. This option is only meaningful when building a dynamic
executable. Audit libraries that are defined within an object with the \fB-P\fR
option typically allow for the auditing of the immediate dependencies of the
object. The \fB-z\fR \fBglobalaudit\fR promotes the auditor to a global
auditor, thus allowing the auditing of all dependencies. See \fIInvoking the
Auditing Interface\fR in \fILinker and Libraries Guide\fR.
.sp
An auditor established with the \fB-P\fR option and the \fB-z\fR
\fBglobalaudit\fR option, is equivalent to the auditor being established with
the \fBLD_AUDIT\fR environment variable. See \fBld.so.1\fR(1).
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBgroupperm\fR | \fBnogroupperm\fR\fR
.ad
.sp .6
.RS 4n
Assigns, or deassigns each dependency that follows to a unique group. The
assignment of a dependency to a group has the same effect as if the dependency
had been built using the \fB-B\fR \fBgroup\fR option.
.RE

.sp
.ne 2
.na
\fB-z\fR \fBguidance\fR[=\fIid1\fR,\fIid2\fR...]
.ad
.sp .6
.RS 4n
Give messages suggesting link-editor features that could improve the resulting
dynamic object.
.LP
Specific classes of suggestion can be silenced by specifying an optional comma s
list of guidance identifiers.
.LP
The current classes of suggestion provided are:

.sp
.ne 2
.na
Enable use of direct binding
.ad
.sp .6
.RS 4n
Suggests that \fB-z direct\fR or \fB-B direct\fR be present prior to any
specified dependency.  This allows predictable symbol binding at runtime.

Can be disabled with \fB-z guidance=nodirect\fR
.RE

.sp
.ne 2
.na
Enable lazy dependency loading
.ad
.sp .6
.RS 4n
Suggests that \fB-z lazyload\fR be present prior to any specified dependency.
```

```
This allows the dynamic object to be loaded more quickly.

Can be disabled with \fB-z guidance=nolazyload\fR.
.RE

.sp
.ne 2
.na
Shared objects should define all their dependencies.
.ad
.sp .6
.RS 4n
Suggests that \fB-z defs\fR be specified on the link-editor command line.
Shared objects that explicitly state all their dependencies behave more
predictably when used.

Can be be disabled with \fB-z guidance=nodefs\fR
.RE

.sp
.ne 2
.na
Version 2 mapfile syntax
.ad
.sp .6
.RS 4n
Suggests that any specified mapfiles use the more readable version 2 syntax.

Can be disabled with \fB-z guidance=nomapfile\fR.
.RE

.sp
.ne 2
.na
Read-only text segment
.ad
.sp .6
.RS 4n
Should any runtime relocations within the text segment exist, suggests that
the object be compiled with position independent code (PIC).  Keeping large
allocatable sections read-only allows them to be shared between processes
using a given shared object.

Can be disabled with \fB-z guidance=notext\fR
.RE

.sp
.ne 2
.na
No unused dependencies
.ad
.sp .6
.RS 4n
Suggests that any dependency not referenced by the resulting dynamic object be
removed from the link-editor command line.

Can be disabled with \fB-z guidance=nounused\fR.
.RE
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBhelp\fR\fR
.ad
.br
```



```
.na
\fB\fB--help\fR\fR
.ad
.sp .6
.RS 4n
Print a summary of the command line options on the standard output and exit.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBignore\fR | \fBrecord\fR\fR
.ad
.sp .6
.RS 4n
Ignores, or records, dynamic dependencies that are not referenced as part of
the link-edit. Ignores, or records, unreferenced \fBELF\fR sections from the
relocatable objects that are read as part of the link-edit. By default,
\fB-z\fR \fBrecord\fR is in effect.
.sp
If an \fBELF\fR section is ignored, the section is eliminated from the output
file being generated. A section is ignored when three conditions are true. The
eliminated section must contribute to an allocatable segment. The eliminated
section must provide no global symbols. No other section from any object that
contributes to the link-edit, must reference an eliminated section.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBinitarray=\fR\fIfunction\fR\fR
.ad
.sp .6
.RS 4n
Appends an entry to the \fB\&.init_array\fR section of the object being built.
If no \fB\&.init_array\fR section is present, a section is created. The new
Appends an entry to the \fB\&.initarray\fR section of the object being built.
If no \fB\&.initarray\fR section is present, a section is created. The new
entry is initialized to point to \fIfunction\fR. See \fIInitialization and
Termination Sections\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBinitfirst\fR\fR
.ad
.sp .6
.RS 4n
Marks the object so that its runtime initialization occurs before the runtime
initialization of any other objects brought into the process at the same time.
In addition, the object runtime finalization occurs after the runtime
finalization of any other objects removed from the process at the same time.
This option is only meaningful when building a shared object.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBinterpose\fR\fR
.ad
.sp .6
.RS 4n
Marks the object as an interposer. At runtime, an object is identified as an
explicit interposer if the object has been tagged using the \fB-z interpose\fR
option. An explicit interposer is also established when an object is loaded
```

using the \fBLD_PRELOAD\fR environment variable. Implicit interposition can occur because of the load order of objects, however, this implicit interposition is unknown to the runtime linker. Explicit interposition can ensure that interposition takes place regardless of the order in which objects are loaded. Explicit interposition also ensures that the runtime linker searches for symbols in any explicit interposers when direct bindings are in effect.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBlazyload\fR | \fBnolazyload\fR\fR
.ad
.sp .6
.RS 4n
Enables or disables the marking of dynamic dependencies to be lazily loaded. Dynamic dependencies which are marked \fBlazyload\fR are not loaded at initial process start-up. These dependencies are delayed until the first binding to the object is made. \fBNote:\fR Lazy loading requires the correct declaration of dependencies, together with associated runpaths for each dynamic object used within a process. See \fILazy Loading of Dynamic Dependencies\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBld32\fR=\fIarg1\fR,\fIarg2\fR,...\fR
.ad
.br
.na
\fB\fB-z\fR \fBld64\fR=\fIarg1\fR,\fIarg2\fR,...\fR
.ad
.sp .6
.RS 4n
The class of the link-editor is affected by the class of the output file being created and by the capabilities of the underlying operating system. The \fB-z\fR \fBld\fR[\fB32\fR|\fB64\fR] options provide a means of defining any link-editor argument. The defined argument is only interpreted, respectively, by the 32-bit class or 64-bit class of the link-editor.
.sp
For example, support libraries are class specific, so the correct class of support library can be ensured using:
.sp
.in +2
.nf
\fBld ... -z ld32=-Saudit32.so.1 -z ld64=-Saudit64.so.1 ...\fR
.fi
.in -2
.sp

The class of link-editor that is invoked is determined from the \fBELF\fR class of the first relocatable file that is seen on the command line. This determination is carried out \fBprior\fR to any \fB-z\fR \fBld\fR[\fB32\fR|\fB64\fR] processing.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBloadfltr\fR\fR
.ad
.sp .6
.RS 4n
Marks a filter to indicate that filtees must be processed immediately at

runtime. Normally, filter processing is delayed until a symbol reference is bound to the filter. The runtime processing of an object that contains this flag mimics that which occurs if the \fBLD_LOADFLTR\fR environment variable is in effect. See the \fBld.so.1\fR(1).
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBmuldefs\fR\fR
.ad
.br
.na
\fB\fB--allow-multiple-definition\fR\fR
.ad
.sp .6
.RS 4n
Allows multiple symbol definitions. By default, multiple symbol definitions that occur between relocatable objects result in a fatal error condition. This option, suppresses the error condition, allowing the first symbol definition to be taken.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnocompstrtab\fR\fR
.ad
.sp .6
.RS 4n
Disables the compression of \fBELF\fR string tables. By default, string compression is applied to \fBSHT_STRTAB\fR sections, and to \fBSHT_PROGBITS\fR sections that have their \fBSHF_MERGE\fR and \fBSHF_STRINGS\fR section flags set.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnodefaultlib\fR\fR
.ad
.sp .6
.RS 4n
Marks the object so that the runtime default library search path, used after any \fBLD_LIBRARY_PATH\fR or runpaths, is ignored. This option implies that all dependencies of the object can be satisfied from its runpath.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnodelete\fR\fR
.ad
.sp .6
.RS 4n
Marks the object as non-deletable at runtime. This mode is similar to adding the object to the process by using \fBdlopen\fR(3C) with the \fBRTLD_NODELETE\fR mode.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnodlopen\fR\fR
.ad
.sp .6

.RS 4n
Marks the object as not available to \fBdlopen\fR(3C), either as the object specified by the \fBdlopen()\fR, or as any form of dependency required by the object specified by the \fBdlopen()\fR. This option is only meaningful when building a shared object.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnodump\fR\fR
.ad
.sp .6
.RS 4n
Marks the object as not available to \fBdldump\fR(3C).
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnoldynsym\fR\fR
.ad
.sp .6
.RS 4n
Prevents the inclusion of a \fB\&.SUNW_ldynsym\fR section in dynamic executables or sharable libraries. The \fB\&.SUNW_ldynsym\fR section augments the \fB\&.dynsym\fR section by providing symbols for local functions. Local function symbols allow debuggers to display local function names in stack traces from stripped programs. Similarly, \fBdladdr\fR(3C) is able to supply more accurate results.
.sp
The \fB-z\fR \fBnoldynsym\fR option also prevents the inclusion of the two symbol sort sections that are related to the \fB\&.SUNW_ldynsym\fR section. The \fB\&.SUNW_dynsymsort\fR section provides sorted access to regular function and variable symbols. The \fB\&.SUNW_dyntlssort\fR section provides sorted access to thread local storage (\fBTLS\fR) variable symbols.
.sp
The \fB\&.SUNW_ldynsym\fR, \fB\&.SUNW_dynsymsort\fR, and \fB\&.SUNW_dyntlssort\fR sections, which becomes part of the allocable text segment of the resulting file, cannot be removed by \fBstrip\fR(1). Therefore, the \fB-z\fR \fBnoldynsym\fR option is the only way to prevent their inclusion. See the \fB-s\fR and \fB-z\fR \fBredlocsym\fR options.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnopartial\fR\fR
.ad
.sp .6
.RS 4n
Partially initialized symbols, that are defined within relocatable object files, are expanded in the output file being generated.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnoversion\fR\fR
.ad
.sp .6
.RS 4n
Does not record any versioning sections. Any version sections or associated \fB\&.dynamic\fR section entries are not generated in the output image.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBnow\fR\fR
.ad
.sp .6
.RS 4n
Marks the object as requiring non-lazy runtime binding. This mode is similar to adding the object to the process by using \fBdlopen\fR(3C) with the \fBRTLD_NOW\fR mode. This mode is also similar to having the \fBLD_BIND_NOW\fR environment variable in effect. See \fBld.so.1\fR(1).
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBorigin\fR\fR
.ad
.sp .6
.RS 4n
Marks the object as requiring immediate \fB$ORIGIN\fR processing at runtime. This option is only maintained for historic compatibility, as the runtime analysis of objects to provide for \fB$ORIGIN\fR processing is now default.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBpreinitarray=\fR\fIfunction\fR\fR
.ad
.sp .6
.RS 4n
Appends an entry to the \fB\&.preinitarray\fR section of the object being built. If no \fB\&.preinitarray\fR section is present, a section is created. The new entry is initialized to point to \fIfunction\fR. See \fIInitialization and Termination Sections\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBredlocsym\fR\fR
.ad
.sp .6
.RS 4n
Eliminates all local symbols except for the \fISECT\fR symbols from the symbol table \fBSHT_SYMTAB\fR. All relocations that refer to local symbols are updated to refer to the corresponding \fISECT\fR symbol. This option allows specialized objects to greatly reduce their symbol table sizes. Eliminated local symbols can reduce the \fB\&.stab*\fR debugging information that is generated using the compiler drivers \fB-g\fR option. See the \fB-s\fR and \fB-z\fR \fBnoldynsym\fR options.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBrelaxreloc\fR\fR
.ad
.sp .6
.RS 4n
\fBld\fR normally issues a fatal error upon encountering a relocation using a symbol that references an eliminated COMDAT section. If \fB-z\fR \fBrelaxreloc\fR is enabled, \fBld\fR instead redirects such relocations to the equivalent symbol in the COMDAT section that was kept. \fB-z\fR \fBrelaxreloc\fR is a specialized option, mainly of interest to compiler

authors, and is not intended for general use.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBrescan-now\fR\fR
.ad
.br
.na
\fB\fB-z\fR \fBrescan\fR\fR
.ad
.sp .6
.RS 4n
These options rescan the archive files that are provided to the link-edit. By default, archives are processed once as the archives appear on the command line. Archives are traditionally specified at the end of the command line so that their symbol definitions resolve any preceding references. However, specifying archives multiple times to satisfy their own interdependencies can be necessary.
.sp
\fB-z\fR \fBrescan-now\fR is a positional option, and is processed by the link-editor immediately when encountered on the command line. All archives seen on the command line up to that point are immediately reprocessed in an attempt to locate additional archive members that resolve symbol references. This archive rescanning is repeated until a pass over the archives occurs in which no new members are extracted.
.sp
\fB-z\fR \fBrescan\fR is a position independent option. The link-editor defers the rescan operation until after it has processed the entire command line, and then initiates a final rescan operation over all archives seen on the command line. The \fB-z\fR \fBrescan\fR operation can interact incorrectly with objects that contain initialization (.init) or finalization (.fini) sections, preventing the code in those sections from running. For this reason, \fB-z\fR \fBrescan\fR is deprecated, and use of \fB-z\fR \fBrescan-now\fR is advised.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBrescan-start\fR ... \fB-z\fR \fBrescan-end\fR\fR
.ad
.br
.na
\fB\fB--start-group\fR ... \fB--end-group\fR\fR
.ad
.br
.na
\fB\fB-(\fR ... \fB-)\fR\fR
.ad
.sp .6
.RS 4n
Defines an archive rescan group. This is a positional construct, and is processed by the link-editor immediately upon encountering the closing delimiter option. Archives found within the group delimiter options are reprocessed as a group in an attempt to locate additional archive members that resolve symbol references. This archive rescanning is repeated until a pass over the archives On the occurs in which no new members are extracted. Archive rescan groups cannot be nested.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBtarget=sparc|x86\fR \fI\fR\fR

.ad
.sp .6
.RS 4n
Specifies the machine type for the output object. Supported targets are Sparc and x86. The 32-bit machine type for the specified target is used unless the \fB-64\fR option is also present, in which case the corresponding 64-bit machine type is used. By default, the machine type of the object being generated is determined from the first \fBELF\fR object processed from the command line. If no objects are specified, the machine type is determined by the first object encountered within the first archive processed from the command line. If there are no objects or archives, the link-editor assumes the native machine. This option is useful when creating an object directly with \fBld\fR whose input is solely from a \fBmapfile\fR. See the \fB-M\fR option. It can also be useful in the rare case of linking entirely from an archive that contains objects of different machine types for which the first object is not of the desired machine type. See \fIThe 32-bit link-editor and 64-bit link-editor\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBtext\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, forces a fatal error if any relocations against non-writable, allocatable sections remain. For historic reasons, this mode is not the default when building an executable or shared object. However, its use is recommended to ensure that the text segment of the dynamic object being built is shareable between multiple running processes. A shared text segment incurs the least relocation overhead when loaded into memory. See \fIPosition-Independent Code\fR in \fILinker and Libraries Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBtextoff\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, allows relocations against all allocatable sections, including non-writable ones. This mode is the default when building a shared object.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBtextwarn\fR\fR
.ad
.sp .6
.RS 4n
In dynamic mode only, lists a warning if any relocations against non-writable, allocatable sections remain. This mode is the default when building an executable.
.RE

.sp
.ne 2
.na
\fB\fB-z\fR \fBverbose\fR\fR
.ad
.sp .6
.RS 4n

```
1576 This option provides additional warning diagnostics during a link-edit.
1577 Presently, this option conveys suspicious use of displacement relocations. This
1578 option also conveys the restricted use of static \fBTLS\fR relocations when
1579 building shared objects. In future, this option might be enhanced to provide
1580 additional diagnostics that are deemed too noisy to be generated by default.
1581 .RE

1583 .sp
1584 .ne 2
1585 .na
1586 \fB\fB-z\fR\fBwrap=\fR\fIsymbol\fR\fR
1587 .ad
1588 .br
1589 .na
1590 \fB\fB-wrap=\fR \fIsymbol\fR\fR
1591 .ad
1592 .br
1593 .na
1594 \fB\fB--wrap=\fR \fIsymbol\fR\fR
1595 .ad
1596 .sp .6
1597 .RS 4n
1598 Rename undefined references to \fIsymbol\fR in order to allow wrapper code to
1599 be linked into the output object without having to modify source code. When
1600 \fB-z wrap\fR is specified, all undefined references to \fIsymbol\fR are
1601 modified to reference \fB__wrap_\fR\fIsymbol\fR, and all references to
1602 \fB__real_\fR\fIsymbol\fR are modified to reference \fIsymbol\fR. The user is
1603 expected to provide an object containing the \fB__wrap_\fR\fIsymbol\fR
1604 function. This wrapper function can call \fB__real_\fR\fIsymbol\fR in order to
1605 reference the actual function being wrapped.
1606 .sp
1607 The following is an example of a wrapper for the \fBmalloc\fR(3C) function:
1608 .sp
1609 .in +2
1610 .nf
1611 void *
1612 __wrap_malloc(size_t c)
1613 {
1614         (void) printf("malloc called with %zu\en", c);
1615         return (__real_malloc(c));
1616 }
```

_____***unchanged_portion_omitted_***