```
**********************************************************
   21866 Wed Aug 20 21:37:17 2014
new/usr/src/uts/common/fs/smbsrv/smb_rename.c
5111 smb_common_rename uses uninitialized variable
Reviewed by: Gordon Ross <gwr@nexenta.com>
**********************************************************
______unchanged_portion_omitted_

 308 /*
 309  * smb_common_rename
 310  *
 311  * Common code for renaming a file.
 312  *
 313  * If the source and destination are identical, we go through all
 314  * the checks but we don't actually do the rename.  If the source
 315  * and destination files differ only in case, we do a case-sensitive
 316  * rename.  Otherwise, we do a full case-insensitive rename.
 317  *
 318  * Returns errno values.
 319  */
 320 static int
 321 smb_common_rename(smb_request_t *sr, smb_fqi_t *src_fqi, smb_fqi_t *dst_fqi)
 322 {
 323         smb_node_t *src_fnode, *src_dnode, *dst_fnode, *dst_dnode;
 324         smb_node_t *tnode;
 325         int rc, count;
 326         DWORD status;
 327         char *new_name, *path;

 329         path = dst_fqi->fq_path.pn_path;

 331         /* Check if attempting to rename a stream - not yet supported */
 332         rc = smb_rename_check_stream(src_fqi, dst_fqi);
 333         if (rc != 0)
 334                 return (rc);

 336         /* The source node may already have been provided */
 337         if (src_fqi->fq_fnode) {
 338                 smb_node_start_crit(src_fqi->fq_fnode, RW_READER);
 339                 smb_node_ref(src_fqi->fq_fnode);
 340                 smb_node_ref(src_fqi->fq_dnode);
 341         } else {
 342                 /* lookup and validate src node */
 343                 rc = smb_rename_lookup_src(sr);
 344                 if (rc != 0)
 345                         return (rc);
 346         }

 348         src_fnode = src_fqi->fq_fnode;
 349         src_dnode = src_fqi->fq_dnode;
 350         tnode = sr->tid_tree->t_snode;

 352         /* Find destination dnode and last_comp */
 353         if (dst_fqi->fq_dnode) {
 354                 smb_node_ref(dst_fqi->fq_dnode);
 355         } else {
 355                 tnode = sr->tid_tree->t_snode;
 356                 rc = smb_pathname_reduce(sr, sr->user_cr, path, tnode, tnode,
 357                     &dst_fqi->fq_dnode, dst_fqi->fq_last_comp);
 358                 if (rc != 0) {
 359                         smb_rename_release_src(sr);
 360                         return (rc);
 361                 }
 362         }

 364         dst_dnode = dst_fqi->fq_dnode;
```



```
 365         new_name = dst_fqi->fq_last_comp;

 367         /* If exact name match in same directory, we're done */
 368         if ((src_dnode == dst_dnode) &&
 369             (strcmp(src_fnode->od_name, new_name) == 0)) {
 370                 smb_rename_release_src(sr);
 371                 smb_node_release(dst_dnode);
 372                 return (0);
 373         }

 375         /* Lookup destination node */
 376         rc = smb_fsop_lookup(sr, sr->user_cr, 0, tnode,
 377             dst_dnode, new_name, &dst_fqi->fq_fnode);

 379         /* If the destination node doesn't already exist, validate new_name. */
 380         if (rc == ENOENT) {
 381                 if (smb_is_invalid_filename(new_name)) {
 382                         smb_rename_release_src(sr);
 383                         smb_node_release(dst_dnode);
 384                         return (EILSEQ); /* NT_STATUS_OBJECT_NAME_INVALID */
 385                 }
 386         }

 388         /*
 389          * Handle case where changing case of the same directory entry.
 390          *
 391          * If we found the dst node in the same directory as the src node,
 392          * and their names differ only in case:
 393          *
 394          * If the tree is case sensitive (or mixed):
 395          *  Do case sensitive lookup to see if exact match exists.
 396          *  If the exact match is the same node as src_node we're done.
 397          *
 398          * If the tree is case insensitive:
 399          *  There is currently no way to tell if the case is different
 400          *  or not, so do the rename (unless the specified new name was
 401          *  mangled).
 402          */
 403         if ((rc == 0) &&
 404             (src_dnode == dst_dnode) &&
 405             (smb_strcasecmp(src_fnode->od_name,
 406             dst_fqi->fq_fnode->od_name, 0) == 0)) {
 407                 smb_node_release(dst_fqi->fq_fnode);
 408                 dst_fqi->fq_fnode = NULL;

 410                 if (smb_tree_has_feature(sr->tid_tree,
 411                     SMB_TREE_NO_CASESENSITIVE)) {
 412                         if (smb_strcasecmp(src_fnode->od_name,
 413                             dst_fqi->fq_last_comp, 0) != 0) {
 414                                 smb_rename_release_src(sr);
 415                                 smb_node_release(dst_dnode);
 416                                 return (0);
 417                         }
 418                 } else {
 419                         rc = smb_fsop_lookup(sr, sr->user_cr,
 420                             SMB_CASE_SENSITIVE, tnode, dst_dnode, new_name,
 421                             &dst_fqi->fq_fnode);

 423                         if ((rc == 0) &&
 424                             (dst_fqi->fq_fnode == src_fnode)) {
 425                                 smb_rename_release_src(sr);
 426                                 smb_node_release(dst_fqi->fq_fnode);
 427                                 smb_node_release(dst_dnode);
 428                                 return (0);
 429                         }
 430                 }
```

```
431         }
433         if ((rc != 0) && (rc != ENOENT)) {
434                 smb_rename_release_src(sr);
435                 smb_node_release(dst_fqi->fq_dnode);
436                 return (rc);
437         }
439         if (dst_fqi->fq_fnode) {
440                 dst_fnode = dst_fqi->fq_fnode;
442                 if (!(sr->arg.dirop.flags && SMB_RENAME_FLAG_OVERWRITE)) {
443                         smb_rename_release_src(sr);
444                         smb_node_release(dst_fnode);
445                         smb_node_release(dst_dnode);
446                         return (EEXIST);
447                 }
449                 (void) smb_oplock_break(sr, dst_fnode,
450                     SMB_OPLOCK_BREAK_TO_NONE | SMB_OPLOCK_BREAK_BATCH);
452                 for (count = 0; count <= 3; count++) {
453                         if (count) {
454                                 smb_node_end_crit(dst_fnode);
455                                 delay(MSEC_TO_TICK(400));
456                         }
458                         smb_node_start_crit(dst_fnode, RW_READER);
459                         status = smb_node_delete_check(dst_fnode);
461                         if (status != NT_STATUS_SHARING_VIOLATION)
462                                 break;
463                 }
465                 if (status != NT_STATUS_SHARING_VIOLATION)
466                         status = smb_range_check(sr, dst_fnode,
467                             0, UINT64_MAX, B_TRUE);
469                 if (status != NT_STATUS_SUCCESS) {
470                         smb_rename_release_src(sr);
471                         smb_node_end_crit(dst_fnode);
472                         smb_node_release(dst_fnode);
473                         smb_node_release(dst_dnode);
474                         return (EACCES);
475                 }
477                 new_name = dst_fnode->od_name;
478         }
480         rc = smb_fsop_rename(sr, sr->user_cr,
481             src_dnode, src_fnode->od_name,
482             dst_dnode, new_name);
484         if (rc == 0) {
485                 /*
486                  * Note that renames in the same directory are normally
487                  * delivered in {old,new} pairs, and clients expect them
488                  * in that order, if both events are delivered.
489                  */
490                 int a_src, a_dst; /* action codes */
491                 if (src_dnode == dst_dnode) {
492                         a_src = FILE_ACTION_RENAMED_OLD_NAME;
493                         a_dst = FILE_ACTION_RENAMED_NEW_NAME;
494                 } else {
495                         a_src = FILE_ACTION_REMOVED;
496                         a_dst = FILE_ACTION_ADDED;
```



```
497                 }
498                 smb_node_notify_change(src_dnode, a_src, src_fnode->od_name);
499                 smb_node_notify_change(dst_dnode, a_dst, new_name);
500         }
502         smb_rename_release_src(sr);
504         if (dst_fqi->fq_fnode) {
505                 smb_node_end_crit(dst_fnode);
506                 smb_node_release(dst_fnode);
507         }
508         smb_node_release(dst_dnode);
510         return (rc);
511 }
```

_____***unchanged_portion_omitted_***