MCR10213
Fix list_dir_info Showing Only the First Name of Saved Entries, and Other save_dir_info Errors
Revision 1.0

Eric Swenson

November 23, 2026

1 Problem Description

Ticket #222 reports that `save_dir_info` mishandles the names of the entries it saves in a `.dir_info` segment. The ticket points at the internal procedure `sseg`, which gets the entry's status with `hcs_$status_long` into the structure at `xxp` but then takes the name count through a different pointer, `p`:

```
            xxp = addr (seg_rec.brstat);
            call hcs_$status_long (dirname, en, 0, xxp, null, ec);
...
            seg_rec.n_names = fixed (p -> branches.nname, 16);
            tp = addr (seg_rec.names);
            do j = 1 to seg_rec.n_names;
               tp -> names (j) = nptr -> names (j+k-1);
               end;
```

and says that `sdir` and `slink` do the same.

1.1 The use of p is correct

`p` is not meant to point at the `status_long` structure. The main loop sets it to the entry's element of the array returned by `hcs_$star_list_ (dirname, "**", 3, ...)`:

```
            p = addr (ep -> entries (i));
            k = fixed (p -> branches.nindex, 18);
            en = nptr -> names (k);
```

The names are copied from the `star_list_` names array (`nptr`), starting at that entry's first name (`k`, from `nindex`), and the count must be the matching `star_list_` count, `nnames`: the number of the

entry's names that match the star name, which for ** is all of them. The local branches declaration matches the first four words of star_list_branch and star_links (star_structures.incl.pl1), and the entries element (bit (144)) is their four-word size. For the directory being saved, p points at brsbuf and nptr at the names its own status_long call returned, again consistently.

So the n_names and names that save_dir_info stores are right, and comp_dir_info, which uses n_names, handles them correctly. Changing p to xxp, as the ticket suggests, would make things worse: as the next section shows, the count at xxp is 0, so no names at all would be saved.

1.2 The actual defect: brstat.nnames is 0

Since MR12.0-1077 (1986, MCR7427), status_ sets status_branch.nnames to 0 whenever it returns no names:

```
/* status_branch.nnames is set to 0 when no names are allocated, regardless of
   whether or not the caller has status permission. As per specs in documentation. */
        status_branch.nnames = 0;
```

sseg and sdir call hcs_$status_long with a null area pointer (they need no names from it), so every segment and directory record is saved with brstat.nnames = 0. Link records are not affected, because slink passes an area to get the link target.

list_dir_info_, which list_dir_info calls for each record, prints the first name and then loops over the rest using brstat.nnames:

```
            call ioa_ ("^anames:^-^a", pfx, seg_rec.names (1).name);
            do i = 2 to binary (seg_rec.brstat.nnames);
```

So list_dir_info has shown only the primary name of every segment and directory for nearly forty years. This is the name miscount the ticket describes.

1.3 Other errors found while reviewing save_dir_info

- **Stale ACL counts.** When hcs_$list_acl, hcs_$list_dir_acl or hcs_$list_dir_inacl fails, nacl keeps the count from the previous entry; sseg and sdir then store that count and copy that many ACL terms from the previous, already freed, list.
- **Leaked link paths.** slink gets each link's target path from status_long, allocated in the system free area, and never frees it. It also frees the names unconditionally, although status_ allocates them only when it returns some.
- **Entry count.** fseg.nents is incremented for every entry listed, including those for which status_long failed and no record was written. comp_dir_info sizes its tables from nents, so this only wastes space, but the count is wrong.
- **Mismatched free.** The initial segment ACL list (a segment_acl array) is freed as a dir_acl array.

2 Proposed Fix

2.1 list_dir_info_: count names with n_names

The three name loops use the record's `n_names` instead of `brstat.nnames`:

```
            do i = 2 to seg_rec.n_names;
...
            do i = 2 to dir_rec.n_names;
...
            do i = 2 to link_rec.n_names;
```

`n_names` has always been the number of names `save_dir_info` stored, so this also lists every name in `.dir_info` segments saved before this change. A comment explains why `brstat.nnames` is not used.

The `compare_ascii` comparison of the current and proposed `list_dir_info_.pl1` is shown below:

```
cpa [lpn list_dir_info_.pl1] ==

Inserted in B:
B45
B46      /* The names are counted by n_names, not brstat.nnames. save_dir_info gets
B47         brstat from hcs_$status_long without an area, and since MR12.0 that
B48         returns nnames = 0 for segments and directories; .dir_info segments
B49         saved before the save_dir_info fix still have it. n_names has always
B50         been the number of names save_dir_info stored. */
B51
Preceding:
A45               if j = 1 then do;


A47                    do i = 2 to binary (seg_rec.brstat.nnames);
Changed by B to:
B54                    do i = 2 to seg_rec.n_names;


A84                    do i = 2 to binary (dir_rec.brstat.nnames);
Changed by B to:
B91                    do i = 2 to dir_rec.n_names;


A137                   do i = 2 to binary (link_rec.brstat.nnames);
Changed by B to:
B144                   do i = 2 to link_rec.n_names;

Comparison finished: 4 differences, 13 lines.
```

2.2 save_dir_info

- `sseg` and `sdir` also store the name count in `brstat.nnames`, so the saved `brstat` is consistent with the record (`seg_rec.brstat.nnames = p -> branches.nname;`); the record for the root directory gets 1. Name handling is otherwise unchanged; a comment before the main loop records why the names and count come from `p` and not from `xxp`.
- After each failing ACL listing call, `nacl` is set to 0.
- `slink` frees the link path when `status_` allocated one (path length greater than 0, otherwise the target is saved as blank), and frees the names only when `brstat.nnames` is nonzero.
- The main loop increments `fseg.nents` only when the entry's record was written (`xp` advanced).
- The initial segment ACL list is freed as `segment_acl`.

The format of the `.dir_info` segment is unchanged, and existing `.dir_info` segments remain valid for both `list_dir_info` and `comp_dir_info`.

The `compare_ascii` comparison of the current and proposed `save_dir_info.pl1` is shown below:

```
Inserted in B:
B62           prev_xp ptr,                                        /* xp before saving an
    entry */
Preceding:
A62           ring fixed bin (3) init (4),


Inserted in B:
B238                    dir_rec.brstat.nnames = bit (bin (1, 16), 16);
Preceding:
A237                    tp = addr (dir_rec.names);


Inserted in B:
B261      /* Each entry's names are copied from the star_list_ names array: p points
B262         at the entry's star_list_ structure (k is the index of its first name, and
B263         nname its name count, which with ** is all of its names). sseg, sdir and
B264         slink therefore take the names from p, not from the status_long structure
B265         at xxp, which has none (see sseg). For the directory being saved, p points
B266         at brsbuf instead, and nptr at the names status_long returned for it. */
B267
Preceding:
A259               dirname = tdir;


Inserted in B:
B276                        prev_xp = xp;
Preceding:
A267                        if j = 1 then call sseg;


A272                        fseg.nents = fseg.nents + 1;
```

```
Changed by B to:
B282                        if xp ^= prev_xp then                          /* count only
    entries actually saved */
B283                            fseg.nents = fseg.nents + 1;


Inserted in B:
B325                    if ec ^= 0 then nacl = 0;                          /* nacl is not set
    on error */
Preceding:
A314                    seg_rec.nacls = nacl;


A321                    seg_rec.n_names = fixed (p -> branches.nname, 16);
Changed by B to:
B333
B334        /* hcs_$status_long was given no area, so it returned no names and set
B335           brstat.nnames to 0 (status_ has done so since MR12.0). Store the real
B336           count there as well, so the record is consistent. */
B337
B338                    seg_rec.n_names = fixed (p -> branches.nname, 16);
B339                    seg_rec.brstat.nnames = p -> branches.nname;


Inserted in B:
B369                    if ec ^= 0 then nacl = 0;                          /* nacl is not set
    on error */
Preceding:
A351                    dir_rec.nacls = nacl;


A369                    if nacl > 0 then free aclp -> dir_acl in (dummy_area);
A370                    call hcs_$list_dir_inacl (dirname, en, areap, aclp, null, nacl,
    ring, ec);
Changed by B to:
B388                    if nacl > 0 then free aclp -> segment_acl in (dummy_area);
B389                    call hcs_$list_dir_inacl (dirname, en, areap, aclp, null, nacl,
    ring, ec);
B390                    if ec ^= 0 then nacl = 0;                          /* nacl is not set
    on error */


A379        gnam:       dir_rec.n_names = fixed (p -> branches.nname, 16);
Changed by B to:
B399        /* As in sseg, brstat.nnames is 0 because status_long returned no names. */
B400
B401        gnam:       dir_rec.n_names = fixed (p -> branches.nname, 16);
B402                    dir_rec.brstat.nnames = p -> branches.nname;


A403                    np = ptr (areap, link_rec.brstat.pnrp);
A404                    link_rec.target = linkpath;
```

```
A405                  np = ptr (areap, link_rec.nrp);
A406                  free np -> names in (dummy_area);
Changed by B to:
B426                  if j > 0 then do;                                  /* status_
    allocates the path only if nonempty */
B427                      np = ptr (areap, link_rec.brstat.pnrp);
B428                      link_rec.target = linkpath;
B429                      free np -> linkpath in (dummy_area); /* it was never freed */
B430                      end;
B431                  else link_rec.target = "";
B432                  if link_rec.brstat.nnames ^= ""b then do; /* and the names only if
    there are any */
B433                      np = ptr (areap, link_rec.nrp);
B434                      free np -> names in (dummy_area);
B435                      end;

Comparison finished: 11 differences, 47 lines.
```

2.3 Retiring the tools instead

The ticket asks whether these tools should be moved to the obsolete library. They are documented commands, `comp_dir_info` is still a useful way to see what changed in a directory, and the fix is small, so this MCR proposes fixing them.

3 Documentation

No documentation changes. `list_dir_info.info` already says that the names of each entry are listed.

4 Test

On a test system:

1. **Reproduce.** In a scratch directory create a segment with three names, a directory with two names, and a link with two names, and add a few ACL terms. With the current software, `save_dir_info` the directory and run `list_dir_info` on the result: only the first name of the segment and directory is listed (both link names are). Keep this `.dir_info` segment.
2. **list_dir_info_ alone.** With the new `list_dir_info_`, list the `.dir_info` segment saved in step 1: all names are listed.
3. **save_dir_info.** With the new `save_dir_info`, save the directory again and list it with `-long`: all names, ACLs, initial ACLs, and the link target are correct. Save `>` as well and list the root record.
4. **comp_dir_info.** Add and delete a name, change an ACL term, delete the link, save again, and `comp_dir_info` the two `.dir_info` segments (old and new): exactly those changes are reported. Also compare the step 1 segment with a new one.

5. **Links.** Save a directory containing a link to a nonexistent target: the link is saved with its target and all its names. (The ACL listing error paths are hard to provoke from a directory the user can list; they were checked by inspection.)

6. **Area use.** Save a directory with many links several times in one process, checking that the system free area does not grow (`area_status`).

5 Files Changed

`bound_dir_info_.s.archive`	`list_dir_info_.pl1`: count names with `n_names`; `save_dir_info.pl1`: `brstat.nnames`, ACL counts on error, link path and names freeing, entry count, `segment_acl` free
`bound_dir_info_.archive`	`list_dir_info_`, `save_dir_info` (recompiled)
`>tools`	`bound_dir_info_` (rebound)

6 Version History

2026-11-23	1.0	Initial version of the MCR.