MCR10131
Fix Race Condition in `install` Resulting in SegFault Condition
Revision 1.1

Eric Swenson

January 26, 2023

1 Problem Description

Once in a while, when performing multiple administrative table installations using the `install` command, the administrator may get a segment fault error. The condition is reported thus:

```
Error: Segment-fault error by install$|2055
(>system_library_standard>bound_install_table_)
referencing 361|0
The segment has been deleted.
r 19:19 6.388 71 level 2
```

The issue is reported in this ticket: https://multics-trac.swenson.org/ticket/302.

2 Analysis

The `install` program makes a copy of the table to be installed in the `>sc1>update` directory. It then updates the header of the table to record some installation options, and then sends a wakeup to the Initializer process to have it perform the actual installation. When the Initializer process handles the wakeup, it installs the table, and deletes the segment in `>sc1>update`. When multiple tables are installed through repeated uses of the `install` program, and scheduling just so happens (usually on a single CPU system) that two (or more) tables are written to the `>sc1>update` before Initializer has completed handling of the first table installation, it is possible for Initializer to delete one of the segments in that directory, while the `install` program is still writing to the segment.

The following sequence of operations may clarify this race condition.

- Administrator calls `install` to install `PDT1.pdt`.
- `install` copies the segment into `>sc1>update`.
- `install` updates the header of the table copy to reflect the working directory, the author, and flags indicating whether authorization and attributes should be updated.

- `install` sends wakeup to Initializer process.
- Administrator calls `install` to install `PDT2.pdt`.
- `install` copies the segment into `>sc1>update`.
- Scheduler preempts Administrator process and schedules Initializer process.
- Initializer process handles the wakeup from the installation of `PDT1.pdt`.
- Initializer uses `hcs_$star_` to look for *all* segments in `>sc1>update`.
- Initializer iterates through all of these segments, installing them and then deleting them.
- Scheduler preempts Initializer process and schedules Administrator process.
- `install` then attempts to update the header of the copy of `PDT2.pdt` (to update the working directory, author, and flags), and gets a segment fault condition because the segment has been deleted.

The issue is that there is no locking mechanism preventing Initializer from processing an in-progress table installation by `install`.

3 Proposed Fix

The proposed fix is both to the `install` program and the `up_sysctl_` program. The latter is the event handler run in the Initializer process for installation requests.

The fix to `install` is to flag the segment it places in the `>sc1>update` directory as "not ready" by adding a well-known suffix. After it has updated the header and set the bit count, it then renames the segment to remove the suffix and terminates the segment.

The fix to `up_sysctl_` is to skip any segments that have the well-known suffix. They will get processed at the next wakeup sent by `install` when it is done processing the segment.

The `install` program names the copy of the table it places in `>sc1>update` with a 15-character unique name formed by calling `unique_chars_`. This results in a segment with a name like "!BBBKWZbCJnXjMx". The proposed fix appends ".temp" to this segment name, and then renames the segment to remove the ".temp" suffix when it is done processing it. It is not possible for the segment name to exceed the maximum entry name size (32), since with the suffix, it will be exactly 20 characters long. There are no other names on the copied table segment, so there are no possible other conflicts.

A `compare_ascii` output of `install.pl1` showing the proposed fix is below:

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

Inserted in B:
B50      dcl  hcs_$chname_seg entry (ptr, char(*), char(*), fixed bin(35));
Preceding:
A50      dcl  hcs_$delentry_seg entry (ptr, fixed bin (35));


A77
Changed by B to:
```

```
B78      dcl  temp_suffix char(5) internal static options(constant) init (".temp"); /*
    suffix added by install while updating tabl
\ce copy */


A248              call hcs_$make_seg (idir, copyname, "", 01010b, copyp, code);
Changed by B to:
B249              call hcs_$make_seg (idir, copyname || temp_suffix, "", 01010b, copyp,
    code);


A259              call terminate_file_ (copyp, bitcount, TERM_FILE_TRUNC_BC_TERM, (0));
A260              copyp = null;
Changed by B to:
B260              /* first set bit count before renaming segment */
B261              call terminate_file_ (copyp, bitcount, TERM_FILE_TRUNC_BC, (0));
B262
B263              call hcs_$chname_seg (copyp, copyname || temp_suffix, copyname, code);
B264              if code ^= 0 then
B265                   call com_err_ (code, "install", "Could not rename table in
    installation directory");
B266
B267              /* now terminate file prior to sending wakeup -- we're done making
    changes to it */
B268              call terminate_file_ (copyp, bitcount, TERM_FILE_TERM, (0));

Comparison finished: 4 differences, 16 lines.
```

A compare_ascii output of up_sysctl_.pl1 showing the proposed fix is below:

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

Inserted in B:
B144     dcl  mailseg_temp_suffix char(5) internal static options(constant) init (".temp");
B145                                                          /* suffix added by install
    while still setting it up */
Preceding:
A144     dcl  status bit (72) aligned;                        /* IOS status */


Inserted in B:
B244
B245                  if index(mailseg, mailseg_temp_suffix) > 0 & after(mailseg,
    mailseg_temp_suffix) = "" then
B246                       go to try_next;                    /* skip this segment until
    it gets renamed */
B247
Preceding:
A242                  call condition_ ("any_other", ucs);     /* Set up a handler in case
    of a fault. */

Comparison finished: 2 differences, 6 lines.
```

4 Documentation

No changes to documentation are necessary since the technical details of table installations has never been documented.

5 Test

I've tested this on an MR12.7 test system (not GHM_Test, which is running post-12.7 code and not GHM, which runs only MCR-approved, installed code). I tested it by doing the following:

I made a version of `install.pl1` with the above changes, but with the call to the rename commented out. Initializer gets the wakeup, finds no segments to install, and logs a message in the answering service log to that effect. Then, I put back the call to rename, and installed ANOTHER table. When AS got the wakeup, it installed just the second table, leaving the first one (not yet renamed) alone. Finally, I manually renamed the first table (got rid of the .temp suffix), and then installed a third table. This time, AS installed the first table first (as it is supposed to do) and then installed the third table. It properly sent a message to my process about all tables it installed.

6 Version History

2023-01-25	1.0	Initial version of the MCR.
2023-01-26	1.1	Integrate reviewer comments – set bc before chname.