Multics Upgrade Installs

Eric Swenson

Contents

1 Introduction

This MTB discusses two approaches that have been used in the past to upgrade an existing Multics system to a new version, and offers the advantages and disadvantages of each. The purpose of the MTB is to engage the Multics community to determine which approach should be used in subsequent Multics releases.

2 Background

Traditionally, Honeywell/Bull produced a set of release tapes that were used both to cold boot a brand new Multics system and to upgrade an existing Multics system to a new release. In both cases, the `reload_system_release` command is used to replace the contents of those directories in the file system (if any) with the contents of those on the release tapes. In the case of a cold boot, of course, new directories and their contents are created. And in the case of an upgrade, the existing directories have their contents replaced, including having segments and directories deleted from the file system if they don't exist on the new release tapes. One final attribute of `reload_system_release` is that the date-time-entry-modified (DTEM) values of existing entries are ignored and entries are replaced even if they are older on the tapes than in the hierarchy.

With MR12.5, which was produced by ACTC, rather than from System-M, a different approach was used. MR12.5 was not delivered with a full set of release tapes. Instead, all the changed segments were collected in an >ldd>MR12.5 directory, and exec_coms were provided in that directory that invoked the `update_seg` tool to install the changed segments in the existing Multics hierarchy. Three separate exec_coms performed the following three actions:

- **install_part1**: Updated hardcore and executable directories
- **install_part2**: Updated >ldd>(source object)
- **cleanup_library**: Cleaned up the temporary segments left by `update_seg` and deleted the >ldd>MR12.5 directory.

In the series of MR12.6 releases (a-e) produced by the DPS8M development team, we used the ACTC approach. In fact, we used the ACTC exec_coms with only minor modifications. However, we actually produced two sets of tapes. The first set was an "update" set that leveraged the ACTC approach. The second set was a full tape set that could be used for cold-booting.

3 Comparison of Approaches

For the purposes of this MTB, we differentiate between the two approaches by using the names "Traditional" and "ACTC".

3.1 Advantages of Traditional Approach

The traditional approach has three primary advantages:

- Only one set of tapes is needed
- It is much simpler to produce

- Guarantees that after an update, the system directories match exactly what was on the originating system

Only one set of tapes is produced and these are used for both upgrade and cold-boot installs. The mechanism used to "load" the tapes is identical in both cases (`reload_system_release`). These tapes can easily be produced from an integration system (e.g. System-M or Gold Hill Multics) by performing an appropriate `backup_dump` with appropriately created dump control files. Both System-M and Gold Hill Multics used the same general approach: As MCRs are approved and audited, they are installed on the system and at some appropriate time, the snapshot of that system hierarchy is used to create the release tapes. This approach is, therefore, quite simple to manage.

A major advantage of the Traditional approach is that after an upgrade release, it is guaranteed that the system directories contain exactly the same contents as those on the system from which the release tapes were created. This helps with being able to reproduce defects because the software environment in which a defect occurs should be the same across all Multics systems at that release.

3.2 Advantages of ACTC Approach

If only an update release is to be made (no cold-boot is possible with an update-only release, as MR12.5 was), the ACTC approach has some distinct advantages:

- A much smaller tape set is required
- Update time is kept to a minimum
- The update can be interactive, allowing for optional updates
- Error recovery is arguably better, with the ability to retry failed updates

Normally a Multics release only impacts a much smaller set of segments than the entire Multics system. In the ACTC approach, only those changed segments are included on the tapes, and thus, the tape set is much smaller than a full set of release tapes required for cold boot. Because the full set of tapes is not loaded with `reload_system_release`, the reload time is much shorter with these update installs.

As an exec_com is used to guide the update process, optional components can be conditionally installed. For example, in the case of the ACTC release, certain separately-priced products were included in the update release. These would only install properly if the package was already installed on the target system. Since the installer was queried as to whether she wanted to install the product, she could answer "no" and skip that portion of the install if the product was not installed on the target system.

Because `update_seg` is used to perform the installs of changed segments, the error checking provided by that tool takes place, allowing the install to be aborted if an inconsistency is detected (a segment to be replaced is missing or has the wrong names, for example). If the exec_com performing the installs runs into an error, the administrator performing the install can examine the exec_com, comment out or fix issues, restart the installation, etc. As it turns out, with the MR12.5 install, ACTC incorrectly predicted the firmware images that might be installed on a given site. It claimed that MR12.5 could be updated from an MR12.3 or MR12.4 system, but actually failed on an MR12.3 system. It was a simple matter to "fix" the exec_com and continue.

Finally, any changes made by the ACTC exec_coms, which simply called `update_seg`, could be reversed due to the way `update_seg` preserves previous versions of segments.

3.3 Disadvantages of Traditional Approach

The Traditional approach has some disadvantages:

- An update release takes much longer to apply
- An update overwrites all segments with those on the tape, trimming any segments not on the tape. (This may actually be a considered a feature).
- The only way to provide optional parts of the update install is to segregate via tape.

The first disadvantage – the length of time to apply update – follows directly from the fact that the entire tape set is reloaded. The tape reload is really all-or-nothing. All files are overwritten and there is no opportunity to provide an interactive, conditional load of certain software packages. The only way to provide conditional install is to segregate separate components onto separate tapes and direct the installer to skip tapes that don't apply. While the fact that `reload_system_release` overwrites all segments/directories and trims any segments not found on the tapes can be seen as a negative, this may actually be a significant advantage because it guarantees the same system directories across Multics systems at that release. Still, for sites like AFDSC, which never actually ran unmodified releases from Honeywell/Bull, this might have been considered a disadvantage.

3.4 Disadvantages of ACTC Approach

The ACTC approach, on the other hand, has a few serious disadvantages:

- If both update and cold-boot are intended to be supported, two sets of release tapes need to be created
- If the update release is intended to allow update from multiple previous Multics versions, the exec_coms can be difficult to author
- The exec_coms need to be tested on each of the system releases to be upgraded
- The source directory for the update tapes is more difficult to keep up-to-date as installs are made on the source Multics integration system

While MR12.5 provided only an update release, the MR12.6 series of releases (a-e) have supported both upgrade and cold-boot installs. This meant that for MR12.6x, two sets of tapes had to be generated and tested. Because a simple directory reload is not used in an upgrade release, and exec_coms must be authored, perhaps carefully crafted with conditional execution depending on what is expected to be already installed on the target system, these exec_coms are difficult to get right. They must be tested on each of the system versions an upgrade of which is supported. As an example, it took the DPS8M development team several tries to get MR12.6d to install on all Multics versions supported (MR12.5, MR12.6a, MR12.6b, and MR12.6c).

Finally, because it is not simply a matter of performing a `backup_dump` of the originating integration system hierarchy, one has to keep track of all the changes to the originating system carefully so that the contents of the upgrade directory (e.g. >ldd>MR12.5 or >ldd>MR12.6) accurately reflect the intended changed components. Simiiarly, the update exec_coms need to be altered each time a new change is installed. Sometimes, the exact same bound component (or source/object archive) may be updated multiple times during a release cycle. In this case, care must be taken in the update exec_coms to not include a component multiple times, or `update_seg` errors will result.

4 Install Times for Both Approaches

In the advantages/disadvantages mentioned above, one difference is the time it takes to perform an update install using the Traditional approach as compared to the ACTC approach. Below are the timings of an update of a MR12.5 system to the latest MR12.6e system. This choice was made because it encompasses an update with as many changed modules as could easily be tested. The first MR12.6a release included a small number of modules. Each successive MR12.6x release included additional modules, such that in MR12.6e, the cumulative set of all changes from MR12.6a to MR12.6e is included.

The test was performed on an Apple Mac Book Pro, with a 2.3 GHz Intel Core i7 processor, 16 GB 1600 MHz DDR3, and a non-SSD HD. It leveraged the standard DPS8M emulator script for the MR12.5-to-MR.12.6e ACTC-style update (MR12.5_to_MR12.6e_upgrade.ini). A separate script (MR12.5_to_MR12.6e_ traditional.ini) was created in order to use the Traditional approach. This script follows the MR12.3 SIB documents for an update install using the `reload_system_release` command to perform the update, as well as an admin mode command sequence to adjust the cutoffs into the PDTs into the 21st century – a necessary step in order to allow daemon logins post 1999. This cutoff adjustment is also found in the ACTC-style upgrade approach.

Neither approach required any operator intervention and both were executed with a single `dps8` command invocation with the appropriate script. The Unix/Linux `time` command was used to perform the timing. The times are shown below:

Approach	**Time**
ACTC	3m52s
Traditional	20m2.436s

Both approaches would take a little longer if human operators had to type the commands manually, with the Traditional approach adding a bit more time due to the greater number of inputs (tape names) required and the consequent delays due to "think time".

While Traditional approach takes about 7 times more time than the ACTC approach, in this particular example, the fact that it only takes about 20 minutes to perform the Traditional update install may lead the reader to consider that the time savings in the ACTC approach is not worth the extra effort involved in preparing this installation approach.

5 Recommendations of the Author

While the purpose of this MTB is to solicit opinions and direction from the Multics community as to the way forward for subsequent Multics releases, since the author of this MTB has very recent experience producing the MR12.6x series of releases, and used the ACTC approach for these releases, it seems fair to allow me to include in this MTB my opinions on the subject.

I created the first MR12.6a release by directly copying the ACTC >ldd>MR12.5 directory and its exec_coms. I made minor changes, as needed to these exec_coms to accommodate features required. For example, because MR12.6 systems were intended to be used with the date/time set in the current century, and because MR12.5 didn't support dates past 1999, all PDTs in MR12.5 systems had cutoffs prior to 1999. Without some special handling, a system upgraded from MR12.5 to MR12.6 would not run because all daemons would not be able to login due to their project cutoffs. I was able to handle this easily by including, in the appropraite update exec_com, logic to update the cutoffs in the PDTs necessary to boot the system in normal mode.

In a later MR12.6 release, it became clear that requiring all users to install MR12.5 and then upgrade to MR12.6 was too inconvenient, so I created full release tapes to allow cold-booting of MR12.6c (and subsequent MR12.6x releases). This resulted in the need to produce two sets of tapes for each release.

Finallly, as a result of trying to support the ability to update from MR12.5, MR12.6a, MR12.6b, MR12.6c, etc. to a later release, the conditional logic in the update exec_coms became prohibitively complex. This was true because I wanted to update only the components that needed updating, regardless of the version of the target system.

After this experience, it became clear to me that the Traditional approach would have been much easier for me. Because the upgrades (and cold boots) were all scripted using DPS8M INI scripts, those users who upgraded didn't have to see the complexity and could update or cold boot with a single DPS8M command line. However, I had to test the upgrade installs on all systems to make sure the exec_coms would do the right thing on each supported target system release.

I'd like to propose that going forward, we go back to the approach Honeywell/Bull used prior to MR12.5 – the Traditional approach, using `reload_system_release` and a single, full set of release tapes. For the issue of conditional installs, if that ever applies in the post Honeywell/Bull Multics world, we can deal with this with separate release tapes. I don't anticipate this being an issue. And in the event that we need to invoke some scripted logic (such as the adjusting of PDT cutoffs in MR12.6a), we can do what Honeywell/Bull would have done: include an exec_com in >tools that performed this functionality, and direct the installer to invoke this exec_com prior to bringing up the system.

The DPS8M development team welcomes any comments on this MTB draft.