MULTICS STAFF BULLETIN MSB - 98

TO: Distribution

FROM: M. J. Grady

DATE: April 30, 1973

SUBJECT: Proposed changes to tssi_

The translator storage system interface module, tssi_, has a number of problems all of which will be aggravated by the installation of Initial ACLs and the removal of CACLs. The purpose of this MSB is to discuss these problems and possible solutions. The discussion will be mostly concerned with the single segment manipulation entries, but similar problems and solutions apply to the multi-segment file entries.

The current implementation is:

1. If the object segment is created by tssi_, it does not save the initial ACL placed on the segment by make_seg, and when the compilation completes it replaces the ACL with one having only one entry, "re" for the creator.

2. If the object segment existed when the compilation started its ACL is saved and replaced with one having one entry, "rwa" for the owner. When the compilation completes tssi_ restores the old ACL. If the restored ACL has only one entry for the owner, it is changed to "re". This was designed as a heuristic technique to catch the case of a compilation which had been aborted and restarted.

3. If the compilation is aborted, tssi_ does nothing except free its allocated storage.

It is clear that there are a number of problems with this scheme. In (1) make_seg will apply the initial ACL to the segment, and it will be lost when the compilation finishes. In (2) the original ACL will be restored but the user will not get "re" unless he is the only person on the ACL. This problem is most evident when a previous compilation is aborted since the segment is left with "rwa" access. Case (2) is touted to offer the slight advantage that ACLs for other users are removed during the compilation. This has dubious value since there are other techniques which the owner of the segment may use which will prevent faults by other users of his segment. Perhaps the worst problem is (3) which will cause an entire, perhaps complex, ACL to be lost.

A relatively simple solution for these problems has been formulated as follows:

1. At the beginning of the compilation the ACL "rwa" for Person.Project.tag will be set. This will be done irrespective of the segment's origin, and irrespective of whether or not an ACL existed already for Person.Project.tag. If the segment was created by tssi_ via a call to make_seg, make_seg will also add "re" for Person.Project.*. If the segment already existed then the only change will be forcing "rwa" for Person.Project.tag.

2. At the end of the compilation the ACL for Person.Project.tag will be removed, again irrespective of its origin. Also if an ACL for Person.Project.* did not exist, the requested mode will be set for Person.Project.*.

3. If the user quits and tssi_ gets control on clean_up then the only action will be to remove the ACL for Person.Project.tag.

The only restriction this scheme imposes is that the instance tag is reserved for use by the translators and if used by other programs, must be reset after each compilation.

This implementation has numerous features, the most vital of which is speed. The entries to manipulate single segment files will involve only three hcs_ calls. Also since the instance tag is used to set the users effective access during the compilation, the rest of the ACL will remain untouched, and will be properly restored at the termination. Note that the ACL entry for Person.Project.* will remain untouched if it existed at the beginning of the compilation. Thus if the user wishes special access to the segment, he may set it via the "*" tag and it will not be changed in subsequent compilations. The entries which create segments will attempt to make a best guess on the final requested mode and call make_seg with that mode. If it is correct, it will save a call to hcs_ at termination.