TO: MTB Distribution

MTB-057

FROM: RR Riedesel, Engineering-PO

SUBJECT: Proposed Modifications to the Multics Encapsulation of the GCOS System

DATE: March 27, 1974

The current version of the GCOS Encapsulation has created a number of accounting and security problems for its host (Multics). This MTB is intended to delineate these problems and to propose modifications to the Encapsulation which will minimize them.

Please send comments to:

RR Riedesel
Honeywell Information Systems Inc.
P O Box 6000 C-61
Phoenix, AZ 85005 (602)993-3868

or:

via "mail" (on Multics System M, Phoenix to

Riedesel. Multics

The addition of the GCOS Daemon to the Multics encapsulation of GCOS has a number of security and accounting problems to its host system. The function of the Daemon is to provide batch (IMCV and card) processing capabilities for the encapsulation. In doing this, it introduces the difficulty inherent in GCOS batch of validating the user for whom the job is being processed. Native GCOS utilizes information extracted from the $IDENT card for accounting identification purposes and information from the $USERID card for file-access (passwording) validation. Unfortunately, the necessity of punching this identifying information on cards makes it unavoidably subject to theft, thereby facilitating impersonation of the GCOS user.

The ease of penetration introduced by the GCOS Daemon is a serious threat to Multics security and accounting. To ensure proper accounting/access control procedures, the following changes to the GCOS encapsulation are proposed.

CURRENT OPERATION

As currently implemented, the GCOS Daemon carries out the following procedures in requesting an absentee job:

1. If a $USERID is encountered in the input stream, the system-master-catalog$password field (col. 15, 12 char.) is used to do a table look-up of a user.project for whom the Daemon will request an absentee job be submitted. (This table is maintained by the GCOS administrator.)

2. If no $USERID is found, the Daemon submits the absentee request in the name of "Absentee.Gcos".

This procedure causes two distinct problems:

1. The accounting system is completely confused, in that any job submitted sans $USERID will have all accruing charges billed to the Gcos project. Multics accounting has no way of routing charges to the actual project for whom the machines resources were expended. Essentially, the Gcos project would have to eat these charges.

2. A security problem inherent in GCOS batch processing is introduced to the Multics system. Any user who is aware of the current implementation can submit non-USERIDed jobs and acquire machine time gratis. Furthermore, if a valid user system-master-catalog$password string is acquired by another user (from a $USERID in a GCOS card deck), he can easily impersonate the user corresponding to that string, thereby gaining access to any of that user's files (whether in Multics or GCOS format). This subverts existing Multics security to an unacceptable extent.

PROPOSED OPERATION

The problems discussed are all intrinsicly related to absentee processes requested by the GCOS Daemon. Interactive users who invoke the GCOS simulator to carry out GCOS activities are validated by Multics passwording techniques at login. It is therefore unnecessary (and probably unwise) to place any further restrictions on the interactive use of the simulator. Instead, it is necessary to ensure that the security of the Multics file system is not derogated by the operation of unvalidated absentee processes that were requested by the GCOS Daemon.

This can be achieved through the following procedures.

1. All GCOS batch jobs processed through the Daemon will have an absentee process requested in the name of "Anonymous.Gcos.g". This will conform to system standards for unvalidated person-id's, and standard Multics access control as applied to this process will, for the most part, deny access wherever it is not warranted. The "g" instance-tag may prove helpful in denying access to these unvalidated absentee processes, while allowing other absentee processes (requested by an interactive user) and all interactive processes running on the GCOS project access as required.

   Thus, by requesting all batch jobs under the process "Anonymous.Gcos.g", the Daemon will prohibit the impersonation of any interactive user. Yet, in a closed-shop environment in which no anonomous users are registered, even this might not be a sufficient plug to the security hole introduced by GCOS batch. Such a situation might exist in a large engineering shop where the firm's product calendar might be kept on-line with "read" access for "*.*.*". Under these circumstances, a non-registered individual (say, for example, a competitor) could submit a card job which, when executing as "Anonymous.Gcos.g", could read the proprietary information from that file. To force an entire user community to set "null" access for the GCOS batch process or "*.*.g" would not only be ungainly, but would also be highly naive in approach, since it would rely upon each individual to carry out this process religiously as each new proprietary data base is created. Therefore, some additional security barrier must be erected between the GCOS batch process and the Multics hierarchy. This is most easily accomplished by using the Multics ring mechanism.

2. Bracket the GCOS simulator [1 5 5]. This would allow interactive processes with execution level (R) = 4 to execute GCOS activities without changing rings. However, any process running with R = 5 which attempts to execute a GCOS job using the simulator will be forced to do so in Ring 5.

3. Register "Anonymous.Gcos" with initial ring of R = 5. This will have the effect of ensuring that all GCOS batch jobs submitted via the Daemon will execute in Ring 5. This will mean that in order for a file to be accessible to "Anonymous.Gcos.g", it will have to be bracketed so as to allow that access for a Ring 5 process. Since there is currently little use made of Ring > Ring 4, and user-created segments are bracketed [4, 4, 4] by default, the majority of >udd will be protected from access by the absentee GCOS process. Yet, it will be quite easy for an interactive user to set the ring brackets and ACL of a given file so as to allow a desired access by "Anonymous.Gcos.g", and any Ring 4 process will be able to access a GCOS batch-created file (bracketed [5 5 5] by default) whenever the ACL so permits.

   Of course, the ultimate in protection would be to run GCOS batch absentees in Ring 7, thereby protecting any user files in all other rings. Unfortunately, this is not currently practical, since the system libraries are bracketed [1 5 5] and any process running in a Ring >Ring 5 would not be allowed access to system subroutines.

These three changes will dissipate the security problems introduced by GCOS batch processing. Although within the portion of the hierarchy accessible to the "Anonymous.Gcos.g" process, there exists a continuing problem of access control, this is almost identical to the situation in native GCOS. The important aspect of this solution is that Multics file security is not affected in a deleterious fashion by the presence of the GCOS environment.

Unfortunately, requiring all batch processes be requested for a single process totally confuses accounting procedures since Multics will bill "Anonymous.Gcos" for all processor time. Therefore, two further changes to the encapsulation are necessary to provide proper accounting.

4. The GCOS Daemon will include a command in the absin file for each absentee request which will set the home-dir of the absentee process to be>gdd>project>person*, where "person.project" are looked up in the User Registration Table (URT). The URT is a data base (maintained by someone at the System Administrator level) which contains a mapping of EPA numbers with corresponding person.project's. The EPA number is

*gdd = gcos_dir_dir

obtained from the $IDENT card by the Daemon and is then used to generate this person and project name to be employed in setting the home_dir.

5. The GCOS simulator will interpret GCOS catalog-file-strings on a $PRMFL card as pathnames with the home-dir of the process as a prefix. Thus, a catalog-file-string RRR$Password/CAT1$Password/CAT2$Password/file1$Password* would be interpreted as home_dir> CAT1 > CAT2 > file1. This means that an interactive user will be able to access PRMFLs under his own home directory, and the absentee batch jobs will be using different subtrees under> gdd for their work, thereby facilitating special GCOS billing routines (to be provided in a later release of the simulator). The interactive Multics user who also submits batch GCOS jobs via the Daemon should, therefore, be allowed "sma" access to>gdd>project>person, so as to permit the sharing of data bases between these two environments. Since his execution level, R, will be 4, he will encounter no difficulties in attempting read or write accesses to files (bracketed [5 5 5]) under this subtree.

   Similarly, when it is necessary for a batch GCOS job to access a file created interactively, the ACL will have to be set appropriately for "Anonymous. Gcos.g", and its brackets will have to be set to [5 5 5] (or higher). This could be done with a copy of the segment if the user would rather not carry out this type of manipulation more than once.

Thus, the first release of the GCOS environment will provide adequate security at the expense of accounting accuracy. However, future releases will include special GCOS billing routines which will further distribute the charges accruing to "Anonymous.Gcos.g" by EPA number.

*The password on each file is optional in GCOS, except for the one on the User Master Catalog--in this example the password on RRR.