To: Distribution

From: Jerry Stern

Subject: Extending the Multics Security Kernel to Ring 1

Date: 06/07/74

This MTB describes some important results of a recent meeting attended by Honeywell, Project MAC, Air Force, and MITRE members of the Multics security group as well as by other interested Honeywell staff members. The primary purpose of the meeting was to explore the question of how system functions (particularly those requiring data bases writeable by all processes) might be implemented outside of ring 0 without adversely affecting system security and certifiability. A new approach permitting such functions to be implemented in ring 1 was agreed upon which represents a major departure from our previous design philosophy.

Prior to the time of the meeting, the notion of a Multics security kernel had been essentially synonymous with the ring 0 supervisor. As discussed in MTB-047, ring 0 was to be the sole interpreter and enforcer of the security controls. Only within ring 0 could a single data segment be permitted to hold information of more than one classification. Outside of ring 0 the security controls would insure that no data segment could be writeable by processes of more than one clearance.

Unfortunately, Multics had not been designed with this particular concept of a ring 0 security kernel in mind. As a result, two distinct problems have arisen. First, non-security related functions have been included in ring 0, thereby making the kernel larger than necessary and consequently more difficult to certify. Second, security-related functions such as inter-user communication (i.e. message segments) have been implemented outside ring 0 in violation of the kernel concept stated above. The second problem is of more immediate concern because it must be resolved as part of the security controls implementation. Therefore, the meeting directed its attention primarily to this point.

Under the security control restrictions, the message segment facility could not continue to operate in its present form. Currently, there exist a number of system message segments (absentee and I/O queues) writeable by all processes in ring 1. As pointed out above, this situation is prohibited by the security controls. Within the confines of our design philosophy, two solutions to this problem were possible. First, the message segment facility could be made a part of the kernel, thereby

giving it the privilege of interpreting security rules. Second, the message segment facility could remain outside the kernel in which case it would be necessary to provide one message segment per process clearance in those cases where a single message segment now serves all processes. Since the number of possible process clearances grows exponentially with the number of categories used (as well as linearly with the number of levels), the second alternative is clearly outrageous if more than just a few levels and categories are used. Therefore, the first alternative was, at least, conceptually preferable.

Unfortunately, under the original design philosophy, moving the message segment facility into the kernel meant moving it into ring 0. This entailed a number of practical problems. For example, if message segments became ring 0 segments, they could not currently be dumped or reloaded because the backup system executes in ring 1. Also, the message segment primitives have not been coded to obey the unwritten laws of ring 0 and hence would have to be reworked. Even if these and other problems could be solved, it was simply undesirable to enlarge ring 0, both from a certification standpoint and from a traditional system design standpoint. Furthermore, the full implication of the problem was much worse than simply a problem with message segments. We were, in effect, committing ourselves to implement all future system functions requiring shared, writeable data bases at least partially within ring 0. This would include, for example, the proposed tape management subsystem.

In the hope of avoiding this objectionable situation, a proposal was made to extend the security kernel to ring 1. It was argued that the choice of ring 1 or ring 0 as the security kernel boundary was essentially arbitrary. The task of certifying the message segment facility would be no less difficult if moved to ring 0. In fact, the isolation between rings 0 and 1 might even facilitate certification to some small extent.

All participants at the meeting seemed to agree with this proposal in principle, although there were reservations expressed as to how it might be put into practice. It was pointed out that if the kernel were extended to ring 1, then non-security related functions which might previously have been placed in ring 1 would now have to be placed in ring 2. Recognizing that a detailed proposal was necessary to clarify such points, the members of the various organizations comprising the Multics security group agreed to work together in developing a general plan for extending the security kernel to ring 1 as well as a particular plan for adding security controls to the message segment facility. These plans will be described in future MTBs as soon as they are finalized.

In the meantime, designers of ring 1 subsystems are encouraged to structure their designs so that security-sensitive code can be easily separated from other code. Security-sensitive code

includes not only code which performs access interpretation, but also code which modifies data segments potentially sharable by processes of more than one clearance. The Honeywell members of the Multics security group are available to discuss any problems of this nature.