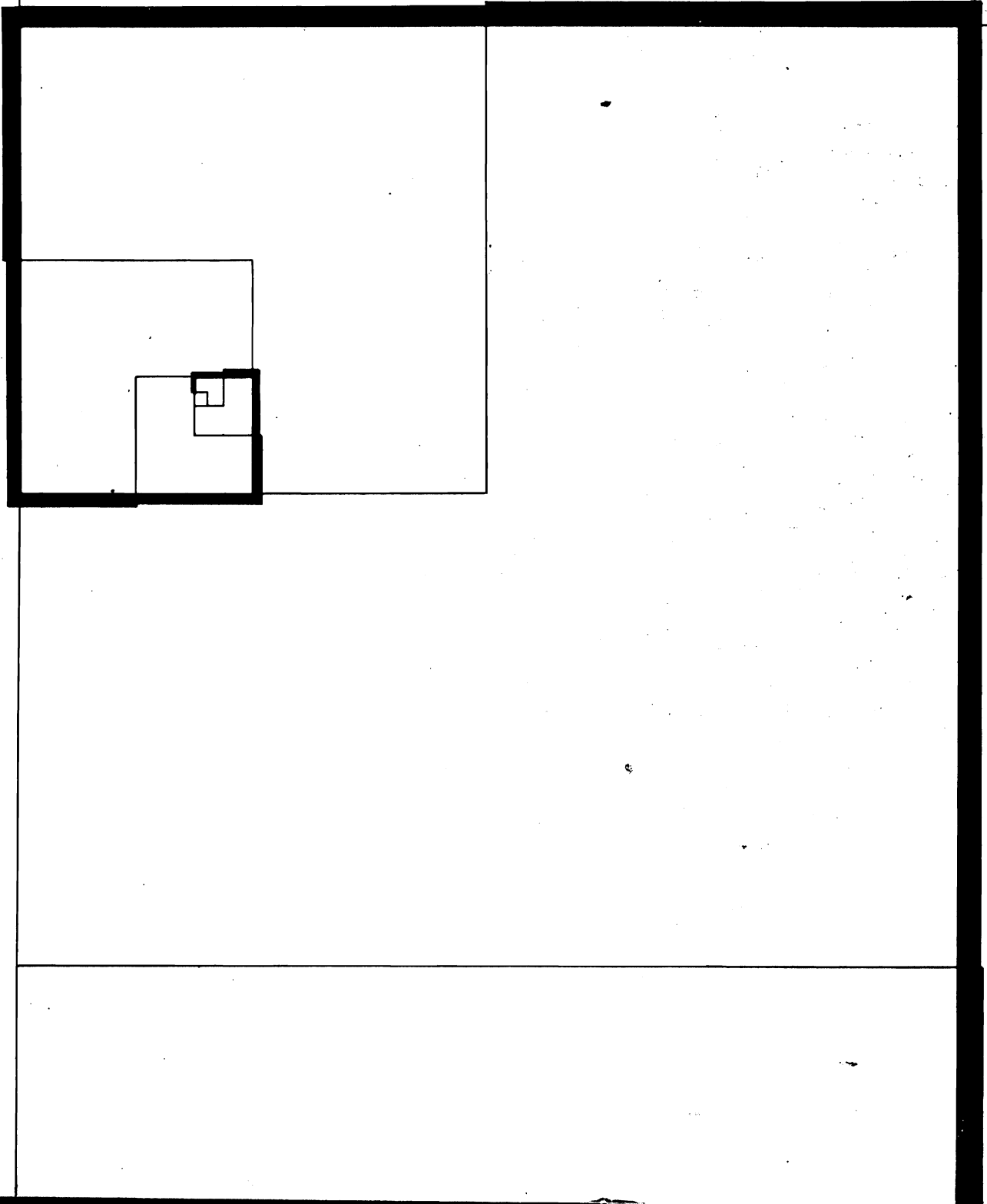


IBM

IBM Data Security Forum

Denver, Colorado

September 1974



In December 1973 the SHARE Security and Data Management Project submitted requirements to IBM in the area of future data security goals. The objective of these requirements was to define an environment within the OS/VS operating system such that a basic security system would exist and implementing a customized security system would be a feasible task for most installations.

IBM has provided a crucial prerequisite of this environment with OS/VS2 Release 2 and its system integrity support. Until this time, the only way to provide a secure system was to limit system access to some secure subsystem. An example of this type of subsystem could be IMS. Unfortunately, the security of such a system leaves much to be desired since most installations, because of economics, are forced to allow application program development and system programming activity to simultaneously occur on the same system as the hopefully secure DB/DC subsystems. These concurrent activities, all occurring on the same data processing system, are an area of considerable concern.

This is not meant to indicate mistrust of the system programmers; rather, this concern is a realization that when there is a breach of system security, those with unrestricted access to the system are, unfortunately, often assumed guilty until proven innocent. We define the datasets which contain the modules of the operating system as an entity called "the system database" and system programmers must deal with this database daily. This database is the most important database on any system, since a violation of its integrity can lead to an undetectable violation of any other database on the system. To treat the system database lightly, or with inadequate security and journalling procedures, is a clear invitation to disaster.

It is my belief that the first objective of a security system must be to protect both the system database and the application program database from unauthorized change. This includes the automatic journalling of sufficient information such that recovery of the system to the state prior to the change is possible, as well as making a record of each change so that accountability rests with a single individual. Furthermore, the journal itself must be completely secure and inviolate. Good maintenance procedures are a byproduct of good security procedures since even unintentional errors can be easily found and corrected.

The ability to identify the responsible party for any

modification to any system, application, or working database on the data processing system is a requirement. To be able to do this implies that the system must be able to identify its users with a reasonable degree of confidence. The degree of confidence should be related to the importance of the database and the level of risk that the installation wishes to assume. This requirement is currently complicated by the fact that today's delivery subsystems such as JES, CICS, IMS and TSO, have no common method of identification and validation of that identification. In fact, some do not provide for any identification at all.

It is for the above reasons that one of the SHARE Security and Data Management Project's requirements called for a common user identifier across all interfaces to the system, and for the ability of the installation to provide its own validation techniques in order to assure the desired degree of confidence in the identification necessary for their environment.

An installation's validation techniques may include the use of passwords, badge readers, or the placement of terminals and other input devices in supervised areas where a secondary visual identification is required by a guard before access is permitted. The required hardware for this support is currently available and the software required is not difficult to produce; yet almost no delivery systems support it. The point is that we have no system-wide user identifier and no software support of the hardware security devices.

It should also be noted that an unsuccessful attempt to access the system may be an indication that the system is under attack. Unsuccessful attempts should be journalled with all available supporting information, including the geographical location of the attempt if it is available. Computer analysis of the journal may provide an installation with information that can help to prevent successful unauthorized access to the system.

Likewise, since the user identifier is the same for all delivery systems and is unique to a single individual, simultaneous access to the system with the same identifier from two terminals is an absolute signal that the security of the system has been breached and security personnel should be immediately dispatched to investigate. Simple computer analysis of all access attempts may also uncover violations of the system security if the accesses from a

single user occur from different geographical locations, and these accesses are physically impossible to accomplish within a short period of time.

Thus, common user identification, sufficient validation techniques, and some method of restricting users to only those subsystems that they need to access are the first lines of defense for any system. This, along with proper journalling techniques, which can limit the responsibility for any action to the level of an individual, which, if well publicized, can act as a deterrent to misuse of authorized privileges, and which can reconstruct the system to the state prior to an undesirable modification, may be all that an installation requires to protect its mission capability.

These procedures are sufficient protection for an installation only under limited circumstances. First, data processing management, either directly or indirectly via corporate management, must have absolute control over all users of the system, such that it would be extremely disadvantageous for someone to misuse his authorized privileges. The second condition is that any unauthorized alteration of data could be quickly found and corrected, and responsibility for this alteration could be easily assigned. The third condition is that the mission capability of the system would not be seriously jeopardized during the interval of time between the alteration and the eventual correction of the data.

The procedures discussed are not adequate if there is no direct control over the users such as in a service bureau environment, if the confidentiality of the data is such that serious consequences would occur if disclosed, or if the correctness of the data at all times is essential to the mission capability of the data processing center.

An additional step towards protecting data would be a program such as IMS which would be supported by the operating system as a single designated interface for accessing a database. This implies that the "designated interface program" must be able to analyze the access request to determine its validity both in terms of the individual making the request and the content of the request. An interface program should also be capable of preventing accidental destruction of the data and be in a good position to efficiently journal changes to the database for later recovery.

However, although designated interface programs are required for system security, to place the authorization decision making within each interface program may be difficult and lead to inconsistencies between various programs. In addition, as was stated previously, if this is to be considered a solution, then the system and application program databases must also be treated securely. The only program and process that is able to protect these databases is the operating system itself, and no preconceived security scheme provided by the vendor will be able to enforce the wide variety of requirements found in various installations.

A good example of this is the question of whether to have centralized or decentralized administration of the security of the data on the system. If the system only services one corporate function, if the corporation believes it is economically and practically feasible to have a central security staff, or if the system handles highly classified information, then a centralized staff for administration of the security function provided by the data processing system is mandatory.

In a service bureau environment, a university, or anywhere that the responsibility for the security and integrity of the data lies outside of the data processing center, then the people who own or are responsible for the data must have administrative control over it. In these kinds of environments decentralized administrative control places security where it is most meaningful.

Any security system must be able to function regardless of whether its administration is centralized or decentralized. It must be a part of both the interface programs and of the operating system if it is to be effective.

In addition, the security system must be fairly efficient. This is not to say that it must induce no overhead, but rather that the overhead induced must not be so great that it is not economically feasible to use the system. Overhead can be reduced by decreasing the number of discrete security decisions that the system must make. It is better to control more global resources such as transactions, rather than to control local resources such as the fields within a database.

The case against controlling local resources can also be strengthened by what can be loosely described as a cross-coupling of resources. An example of resource coupling

would be that, while it may be permissible for a user to modify a field in a database, a condition imposed may be that some other field is also modified in a predetermined manner. This would require a sophisticated designated interface program or simple transaction level authorization. In this case, control at the transaction level would reduce overhead, simplify the analysis of the system, and not jeopardize the security of the data.

Unfortunately, as the magnitude of the resources being controlled increases, the proportion of responsibility for the implementation of the security system shifts from the vendor to the customer. For example, while the responsibility for the limitations of access to specific transactions may lie with the vendor, what the transaction does and what rules it follows are the customer's responsibility.

If the data and programs residing in a data processing system are considered the resources of that system, then the function of the security system is to control the interaction between the users of the system and these resources, in a manner predetermined by the administrators of the resources.

Whatever the implementation of this system, it must be easy to use and easy to understand. For many systems, a high requirement is that it be easy to audit. Complexity and confusion within the system may make the auditing process difficult and therefore lead to unknown irregularities.

However, an uncoordinated approach, with a multiplicity of resource control routines in the various interface programs, in the operating system, and in other miscellaneous application programs, is neither easy to use nor easy to understand. Understanding the complex interaction for several systems, each with its own unique database and algorithmic processes, will make comprehension of the system as a whole difficult at best. Finally, demonstrating consistent application of resource control will be time consuming and difficult.

On the other hand, a single process within the system, making resource control decisions, must treat resources consistently and be easily extendable for new applications. Its interfaces to the system should be modifiable so that, with simulation, its decision making processes could be more easily tested, understood and verified. With a well planned

set of interfaces via the system control program, it would be easy to use for application programs. Since it would be removed from the application programs themselves, application programmers need not know the exact decision making process that would be used. Conversely the decision process could be easily modified without having to modify each of the application programs. And finally, since it would be removed from the physical resource control, it could easily control conceptual resources such as program paths.

Examples of program paths are transactions, command sequences, and operating system processes such as "open". A program path can also be defined to include the flow of control within a module. This enables an installation to define different security levels for different paths within an application program, without having to rewrite different application programs due to the differing requirements for security. It also avoids the need for coding authorization checks within the application program itself.

All authorization decisions would be made within a central control facility which would be a service function of the operating system. The central facility for resource control would make decisions when requested by other parts of the operating system or application programs. Implementation of the decision would be left to the requestor. Possible responses by the central facility could include a yes or no answer, a limit on a quantitative resource, an error message to be displayed to the user, or the name of an error routine to which control should be transferred.

The title of this paper, Centralized Resource Control Information Facility, is the title of a SHARE Security and Data Management Project requirement which states:

"There should be a centralized bank of resource control information and an installation replaceable operating system provided service for accessing and maintaining it. The resource control information must relate resources (such as datasets, program paths, etc.), conditions under which they can be made available (such as level of validation), and user identifiers. New types of resources, the resources themselves, the conditions, and the user identifiers must all be installation definable.

All authorization and delegation must flow through the

single operating system access and maintenance service, and this service must be invocable during normal production operation. Invocation for the purpose of validating access to a resource should return a yes or no answer and optionally a variable length byte string to be used in corrective action (e.g. an error message, a module name, or a limit on a quantitative resource). Security violation recovery routines should be modular and designed for easy user replacement.

Both algorithmic grouping and grouping by itemization for both resources and user identifiers should be possible."

The incentive given for this requirement was that "demonstrable consistent application of resource control has become a requirement." In addition it stated that administration of resource control will be facilitated by its centralization within the data processing system.

Having a central facility such as that described above, means that more effort can be spent in making it a sophisticated product. And sophisticated it must be, if it is to be useful. Algorithmic grouping of resources is an example of this sophistication. In addition to itemizing resources, the requirement states that resources must be able to fit into groups as described by some algorithm. The TSO notation of asterisk for dataset naming conventions to indicate all qualifiers as in Userid.PROGRAM.* is an example of this. This means that the security information database need not be updated each time a dataset is created or deleted and will automatically assume whatever level of protection that is specified by the algorithmic group that describes it.

The implications of the system described are interesting. Security is a joint customer-vendor endeavor. It is the responsibility of the vendor to supply the services within the operating system to access the centralized resource control information facility; to provide an easily extendable set of routines that comprise the facility for the basic decision making of the operating system; and finally to provide the capability for performing maintenance on the information used by the security system. With vendor supplied delivery systems, such as IMS or CICS, security support should be provided as additions to the central security facility rather than as independent routines. And

finally, support for a system-wide user identifier and an easy-to-use journalling facility are required.

It is the installation's responsibility, however, to verify that their own application programs formulate requests correctly to the facility and take the correct action as specified in the facility's response. This means that application programs should formulate their requests to the central security facility in a manner consistent with the operating system's use of this facility. This is an obvious example of the consistency of application criterion for security rules.

Although it is not entirely satisfying to have a security system that is neither totally the vendor's responsibility nor the customer's, this set of recommendations seems to offer the best base for continued extendability. We must acknowledge that this is an era when systems are constantly being used for new applications at the same time that there is constant pressure for consolidation of facilities and databases. These pressures are in turn forcing less specialized and more general purpose operations.

The concept of the centralized resource control information facility, along with a system-wide user identifier and proper journalling techniques, is capable of simultaneously satisfying the requirements of a batch system, an interactive timesharing system, and a data base/data communications system. It is time to develop a coordinated system-wide approach to the solution of our resource control problem.