

SHARE VS/OS Security and Data Management Project

Goals for Data Security

The SHARE VS/OS Security and Data Management Project is holding this open session in order to acquaint the general SHARE membership with our goals for data security.

In San Diego, at the December 1972 interum SHARE meeting, this project started its investigation into the problems of data security. The group attending that meeting had very diverse representation, being composed of representatives from educational institutions, service bureaus, industry and Department of Defense installations. As the discussions progressed, we arrived at two conclusions:

- * None of us were satisfied with the non-existent level of security provided by OS/MVT.

- * We could not reconcile the different requirements of the group.

As I look back at it, the main problem we had was separating the issues of system integrity from data security.

Despite these divergent viewpoints we did agree on certain requirements of a security system. These are:

- * The security system should be part of the operating system - not an add-on package

- * Identification and validation of users is the first level of security

- * The security system should not be able to be turned on and off by an "authorized user" - it should be continuously active

- * The system should be able to run a highly secure job without having to purge itself of all other jobs or users

- * The security system should be able to selectively invoke high overhead functions, such as dataset purging or rewriting with zeros, on an individual resource basis

- * An interface program, for example IMS, should be

supported as the only way to access a specific dataset

A short time later, IBM announced its forty million dollar program in data security. Then they announced VS2 Release 2, which provides the basis for a secure system - integrity. Suddenly the project was freed of the constraints of worrying about both system integrity and data security, and could focus its attention solely on data security.

Soon IBM will ship VS2 Release 2. For the first time a general purpose operating system that is guaranteed to have integrity will be available to the IBM user community. To quote the VS2 Release 2 Planning Guide's section on System Integrity:

"A highly desirable property of an operating system is its ability to insure that one program cannot interfere with or modify another program's (system or user) execution unless it is authorized to do so. For reliability and system availability, this is extremely important; for data security, it is essential. VS2 Release 2 provides this capability in the form of system integrity support.

System integrity is defined as the ability of the system to protect itself against unauthorized user access to the extent that the security controls cannot be compromised. That is, there is no way for an unauthorized program using any system interface to:

- * bypass store or fetch protection, i.e. read or write from or to another user's areas.

- * bypass password checking, i.e. access password protected data for which a password has not been supplied.

- * obtain control in an authorized state.

In VS2 Release 2 all known integrity exposures have been removed. IBM will accept as valid, any APAR that describes an unauthorized program's use of any system interface (defined or undefined) to bypass store or fetch protection, to bypass password checking, or to obtain control in an authorized state."

I assume that the severity level of these APAR's can range

from severe to trivial, just like normal APARs, depending on the severity for the installation involved. However, handling of the APAR's cannot be done normally. Placing the APAR in Early Warning Microfiche will jeopardize the security of every installation running the system. Yet to not notify the installations will leave them in a position of vulnerability. This is indeed a serious problem and we have not resolved it as yet.

Therefore, with the integrity commitment from IBM, it is now is an opportune time for us to develop our set of data security requirements for an IBM operating system.

Although physical security, protection from destruction of data through a natural or man-made disasters, disclosure of data via electronic evesdropping, etc., are crucial issues in the total security picture, we feel that security systems from the operating system point of view are now an overriding point of concern. We are well on our way to solving the other issues, at least to the point where the risks associated with them are negligible compared to operating system security.

Pressure is also being brought to bear on us from other directions. More and more states, and the federal government as well, are enacting laws to prosecute those who disclose confidential personal data even if it is by negligence.

In addition, as we continue to consolidate and centralize our data processing centers, and with the trend towards larger online databases, we increase our risks of accidental or malicious destruction and alteration of data. In many cases, we have dropped our manual backup systems. Even if we did have them, we have become highly dependent on computer based systems. For example, many companies have used the capabilities of computers to reduce their inventories to a minimum. Reverting to a manual system, even with accurate local inventories, could not be accomplished without a substantial period of chaos.

A few years ago, one could assume that someone might attack a database for personal gain. Unfortunately, nowadays, large corporations and governmental agencies may have their systems attacked solely for the purpose of disrupting service, with the attacker not really caring whether he is caught. Smaller companies are not immune either, since a revengeful employee could be similarly motivated.

We can no longer be complacent with the knowledge that we can probably apprehend all violators; because even fear of

apprehension is not always a deterrent. The motivation for the attack may be to interrupt service and have nothing to do with personal gain.

We must have adequate identification techniques and journaling systems that can allow us to reconstruct what happened. We must also have authorization and surveillance systems that will halt any attack at its early stages - before it can do any damage. Once the security system recognizes that an attack is in progress, it must take appropriate action to protect itself. Furthermore, the security system must be capable of tracking attacks which have failed and of initiating affirmative action to prevent further attacks from this source.

This action is obviously dependent on the installation and the data involved; but it may range from just notifying security personnel to actually shutting down the computer system itself. Whatever the case, the system must act reliably and predictably and must fail-soft, that is, if the system is to fail, it must fail in a predictable manner.

We believe that security must be a joint vendor-customer endeavor in that the operating system and all subsystems and application programs must work together in providing defense of data. Thus installations must develop security standards for all application software and provide mechanisms to enforce these standards.

Many times the argument is encountered that some other computer vendor has produced a security system that is independent of the application programs. Where this is true, it is generally applicable only to single terminal/task relationships, introduces restrictions on sharing of data, and would require rebuilding our existing systems from the bottom-up. Multiple terminal/task systems, where the terminals have different security requirements, still require the cooperation of the application program to enforce security. Also, unfortunately, we have been developing our systems based on an operating system that was designed in the early 1960's. And these designers just did not conceive of the applications and the associated security requirements of the systems that are in use today.

Most of us here have too much invested in application systems to be able to afford to start over and redesign our systems to run under a totally new operating system. Besides, many data dependent security constraints still could not be enforced without some security in the application program itself.

It is necessary to provide for auditability and administration of both applications and the system itself. Placing security controls in these areas, each independent of the other, could lead to obvious and severe inconsistencies. Consequently, in VS/OS Group Requirement #73-86, we have recommended the following:

Description:

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. Invokation 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.

Incentive:

Demonstrable consistent application of resource control has become a requirement. In addition, administration of resource control will be facilitated by its centralization.

This facility, along with security standards in all "secure" application programs to make sure that they invoke it, solves the problems of consistency of application, ease of administration, and auditability.

Notice that in addition to physical resources such as

datasets, the facility handles logical resources such as program paths and anything else the installation wishes to define. Thus an installation's security management will be able to say that an individual is able to enter a certain type of transaction from a set of designated terminals between certain hours. In this case, the transaction type can be considered the resource and the latter statements are the conditions under which the individual can access the resource.

The requirement also means that the information facility must be able to dynamically update its databank while the system is running. The system cannot be required to be unavailable during entry or processing of these changes.

Algorithmic grouping of resources and user identifiers is required; for example a group of datasets can be referred to as "SYS1.*". This means that, as datasets are added or removed from the system with the names beginning with "SYS1", they will automatically attain the protection level of the group without costly database updating for each addition and removal. This is particularly important in interactive environments where datasets are created and deleted continuously. The user and the installation must be able to specify global security requirements for these resources.

Along with this facility, an authorization system or application program is necessary to update the database. This program must be able to relate the system's secured resources to an administrator in the administrator's language. It must be easy to use by non-technicians, such as those responsible for defining authorization privileges.

For "open shop" installations, it is necessary to introduce the concept of the "owner" of the resource where resources may be datasets and programs. The "owner" is the administrator of that resource and may specify who can access it. He may also request that accesses be journaled. This is particularly important when dealing with proprietary programs and databases.

All of the five requirements submitted to the project use the phrase "installation replaceable." We define "installation replaceable" to include "easily modifiable in its source language." Thus, the default routines provided by IBM should be designed for ease of modification, well documented, and written in a language for which we have a compiler.

A phrase that is often used in security systems is "acceptable level of risk." We believe that using VS2 Release 2 as a base, the above requirement provides the authorization and delegation functions which are mandatory for a secure system and that installation security standards aided by the other project requirements submitted at the same time can allow an installation to maintain whatever it considers to be its acceptable level of risk.

We feel that an installation is never totally secure and always has some chance of an attack against it succeeding. By devoting effort in the proper areas, an installation can reduce the probability of an attack succeeding, or can lower its level of risk.

We have determined five areas where effort must be placed to reduce the level of risk. Four additional requirements were submitted to IBM to give us the proper tools within the operating system to increase our security. One area is solely the installation's responsibility. These areas are:

- 1) installation security standards
- 2) level of identification and validation of each user
- 3) journaling facility and recovery procedures
- 4) security violation recovery
- 5) designated interface programs

Installation security standards:

There is no way to underestimate the importance of adequate security standards and their strict enforcement. The programmers who code the application programs, the systems programmers who maintain and customize the operating system, the security staff that maintains and customizes the security violation and validation routines, all have an opportunity to subvert the security of the system.

As programmers and managers, we should welcome strictly enforced security standards and complete journaling of all operating system and application program changes. Systems programmers are in a particularly precarious position since they can be blamed for almost any incident on the system. Being able to affix responsibility for some action to an individual also means it is easy to exonerate everyone else.

However, we do not expect that this mechanism will get its greatest use from tracking down security violators, but rather that errors in the system will be more easily determined and corrected. Since all changes to application programs and to the operating system must be journaled in detail, they can easily be backed off. Improved maintenance procedures are a significant by-product of security standards.

The minutes of the project meeting last August reflect this concern:

"An important point was raised about the system database (e.g. LINKLIB, LPALIB, etc.). This is one of the most critical databases on any system and a full audit trail is absolutely necessary both in case of eventual security violations and standard recovery in case of system failure."

IBM, in a recent announcement, has given us a tool called the System Modification Program which is due to be shipped at the end of this month. This service aid applies superzaps, module replacements, macro replacements, and source code changes via IEBUPDTE. It keeps a detailed log of all changes and provides a back-off capability for removing modifications.

Level of identification and validation:

Another area of risk in a secure system is "How sure is the system that a user is who he says he is?" This problem is compounded by the use of different identification and validation procedures in batch, TSO, APL, IMS, etc.. Group requirement #73-85 aids an installation in reducing its risk in this area:

Description:

All subsystems must call upon a single installation replaceable subroutine for the purpose of identification and validation of all users of the system. A default routine should be provided by IBM for this purpose along with full specifications of the interface requirements.

The subsystem should provide to the subroutine the origin of the entry request and full facilities for interacting with the user and related equipment.

Incentive:

Identification and validation of the user is a prerequisite to any data access control system.

Comment:

The purpose of this requirement is not to ensure identical logon protocols but rather to specify that the information collected and the validation procedures are identical for both batch and interactive systems.

This requirement gives the installation the ability to validate the identity of the user to whatever level it wishes and to use this level of validation in later security access decisions. For instance, with hardware now readily available, it is possible to have the following levels of validation:

- * Password only

- * Password/Badge reader

- * Password/ 2 Badge readers - one being used by a guard who has visually identified the user.

More exotic validation techniques could use fingerprints,

voice prints, lip prints, etc.. Identification and validation of a user's identity is the cornerstone of security. This, combined with proper journaling, provides the ability to limit the accountability for some action to an individual. It is important to think of the initial identification and validation procedures as being the first line of defense of the system. The risk of data security violations is reduced by limiting access to the system to only those who are authorized to use it.

Journaling facility and recovery procedures:

Application designers must be encouraged to journal everything that is necessary to reconstruct an event or recover a database. Journaling should be a service offered by the operating system just like the access methods. Depending on the installation, the journaling facility may have to handle very large volumes of information and utilize several output devices. It may have to have the strictest integrity and use special hardware such as a special tape drive and controller that buffers the output within the controller so no data can be lost by a system failure. Another possible hardware requirement is a "write once" tape that cannot be rewritten.

The main requirements of a journaling facility therefore are:

- * easy to use
- * able to handle large volumes of data efficiently
- * be tailorable to an installation's needs
- * provide an unimpeachable record of activity for auditability and recovery

VS/OS Group Requirement #73-87 requests this facility:

Description:

There should be a centralized installation replaceable journaling facility provided by the system control program which can be invoked by any program. This facility must be capable of recording on a variety of data storage devices. Where IBM provided subsystems use this facility, programs should be provided by IBM to analyze these entries.

Incentive:

A centralized journaling facility is necessary for auditing and recovery.

Another area of concern is a misuse of authorization. For example, a person may be authorized to access certain types of services, but by combining them in an unforeseen manner, he would be able to do something that is not authorized.

Detailed journaling by the application programs and by certain operating system interfaces would allow reconstruction of the sequence of events which led to the unauthorized use.

Security Violation Recovery:

It is admitted by most security people that, given enough time, any security system can be violated. However, in the process of testing a system's defenses and attempting to bypass them, indications of these attempts will occur. Whether the system will actually be violated, is therefore a function of how alert an installation's security staff is, how good the installation's security violation analysis programs are, how well attuned the security system is to the installation's needs, and what action the system takes to preserve its security.

A security violation facility should be provided by the operating system in order to institute timely and appropriate recovery action. Group requirement # 73-88 addresses this need:

Description:

The system control program should provide an installation replaceable security violation exit. An IBM default routine should be provided which makes a journal entry and writes a message with a security violation routing code.

Incentive:

Timely corrective action and notification of the proper installation personnel is critical. This is an extremely installation dependent function.

A trivial example of this requirement is the case of several failures on a prompt for a password at logon. Most systems merely disconnect the line - a secure system should call upon the security violation exit which will deactivate the user-id until security personnel have investigated.

Designated interface programs:

The project's final requirement deals with designated interface programs. We do not feel that it is the responsibility of the operating system to do "record and field" level data protection although we do feel that the operating system should maintain the authorization database which is involved in making those decisions.

The possibility of an application program inadvertently modifying data incorrectly and the chance that an authorized user of a database may bypass security and journaling functions by using his own program to access a database led us to group requirement #73-89:

Description:

There should be the capability of associating with any dataset a single interface program capable of accessing that dataset. Where the interface program is a subsystem (e.g. IMS) an interface should be provided to other subsystems (e.g. TSO).

Incentive:

The need to be able to limit the path to a dataset to one interface program structures the system so as to provide increased integrity, security, and back-up capabilities.

This requirement also addresses the problem of data integrity. Data integrity is concerned with the accuracy and completeness of the data while data security is concerned with protection from unauthorized destruction, modification, or disclosure whether accidental or intentional. Misuse of authorized privileges, errors in application programs, etc. can cause a database to lose its integrity. In these cases data security, as we define it, has not been violated.

A designated interface program is in a position to perform validation of database update requests and to journal all changes. This helps to protect the integrity of the database by providing a record of database modifications for error determination and recovery. In addition, it can be much more efficient than the operating system in journaling these changes since it is able to deal with specific and meaningful information. The operating system would have to journal all data changes, thereby incurring much more overhead.

In summary, VS2 Release 2 provides a base for a data security system because of its integrity features. The VS/OS Security and Data Management Project has determined the following requirements for a data security system:

- * Security system should be an integral part of the operating system and should be tailorable to an installation's needs.
- * High overhead security functions should be selectively invoked on an individual resource basis.
- * Users should be identified and validated in the same manner at all interfaces to the system.
- * Resource control information and decision making should be centralized.
- * Journaling capability should be an operating system service.
- * Security violation handling should be centralized.
- * Designated interface programs should be supported on a dataset basis.

The five requirements submitted by the project on this issue were:

- * 73-85 Identification and Validation Exit
- * 73-86 Central Resource Control Information Facility
- * 73-87 Centralized Journaling Facility
- * 73-88 Centralized Security Violation Exit
- * 73-89 Designated Interface Programs

If anyone has any questions, opinions, or suggestions concerning these topics or any other aspect of data security, please contact me or any of the other project members.

Barry Schragar UIC
Project Manager
March 4, 1974