AD/A-007 059

SECURE MULTILEVEL VIRTUAL COMPUTER SYSTEMS

R. Rhode

Mitre Corporation

Prepared for:

Electronic Systems Division

February 1975

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

Mark H. Richardson

MARK H. RICHARDSON, 1Lt, USAF
Project Engineer

Roger R. Schell

ROGER R. SCHELL, Major, USAF
Project Engineer

FOR THE COMMANDER

Robert W O'Keefe

ROBERT W. O'KEEFE, Colonel, USAF
Director, Information Systems
Technology Applications Office
Deputy for Command & Management Systems

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
ESD-TR-74-370

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER
AD/A-007 059

4. TITLE (and Subtitle)
SECURE MULTILEVEL VIRTUAL COMPUTER SYSTEMS

5. TYPE OF REPORT & PERIOD COVERED

6. PERFORMING ORG. REPORT NUMBER
MTR-2890

7. AUTHOR(s)
R. Rhode

8. CONTRACT OR GRANT NUMBER(s)
F19628-73-C-0001

9. PERFORMING ORGANIZATION NAME AND ADDRESS
The MITRE Corporation
Box 208
Bedford, MA 01730

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
Project No. 572R

11. CONTROLLING OFFICE NAME AND ADDRESS
Deputy for Command and Management Systems
Electronic Systems Division, AFSC
Hanscom Air Force Base, Bedford, MA 01731

12. REPORT DATE
FEBRUARY 1975

13. NUMBER OF PAGES
33

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Multilevel Security
Security Kernel
Virtual Machine Monitor
Virtual Machines

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
In order to securely process data at varying levels of sensitivity, the users and data of the various levels are, at present, physically isolated from those at other levels. This paper presents an approach to insuring the isolation of these levels through the use of the logical controls present in a virtual computer system (VCS). In addition, possible uses of a secure multilevel VCS are suggested and parallels are drawn with secure computer system development that would aid in the certification of the VCS security controls.

TABLF OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION I

INTRODUCTION

Multilevel computer facilities, i.e., those that securely process data at several levels of sensitivity, are quite common in the military, governmental and civilian areas. These installations generally provide inter-level security by running users and data of a given level on machines that are temporarily or permanently dedicated to that level. Therefore, in order to accommodate more than one level it is necessary to either:

- maintain multiple copies of the hardware; or
- employ job scheduling to dedicate a set of hardware to several levels throughout the day.

These spatial and temporal techniques of providing physical isolation for individual levels afford a high degree of security but can cause inefficient use of the hardware and make use of the facility inconvenient.

This paper discusses the use of virtual machine architectures as a means of providing multilevel service of essentially the same type as described above*, but with several important advantages.

Section II of this paper provides a general background in virtual computer systems. The general application of these systems to providing multilevel service is introduced in Section III and an approach to proving system security is suggested. Section IV deals with the problems and tradeoffs that might be encountered when considering a virtual computer system approach to multilevel service. Finally, Section V presents specific applications in which a multilevel virtual computer system might be useful.

*That is, with complete partitioning of levels. Unless otherwise stated, then, the term multilevel security in this paper refers to the maintenance of a total separation of the security levels.

SECTION II

VIRTUAL COMPUTER SYSTEMS

INTRODUCTION

Through the use of multiprogramming, many operating systems are able to support several users concurrently. This capability is achieved by allowing the operating system (OS) to control the hardware resources and multiplex them among the users. In effect, then, the OS is giving each user the illusion that he is running on his own machine. That is, the OS creates individual environments in which the user's code executes (Figure 1).[1,2,3]

In reality, massive amounts of OS software are not directly related to multiprogramming but rather are used to provide sophisticated services to the users. Therefore, the user environments may possess not only the basic hardware resources (CPU, I/O devices, core) but also such complex resources as a hierarchical file system, I/O managers, access methods, etc. For this reason, the user environments of an OS are referred to as extended machines.

VIRTUAL MACHINE ORGANIZATION

The virtual computer system (VCS), as seen in Figure 2, has an organization that strongly resembles that present in the general computer system. The basis of a VCS is an OS like, software nucleus called a virtual machine monitor (VMM) which establishes distinct user environments called virtual machines (VM's). These virtual machines possess none of the sophisticated resources supplied by an OS to its extended machines but instead strongly resemble the host hardware environment. In general, the VM is patterned after the hardware interface produced by a computer in the same family as the host computer and a sufficiently versatile VMM may concurrently support virtual machines resembling several different family members.

The utility of the VCS is derived from the fact that the VM environment is capable of supporting any operating system designed to be run on the corresponding hardware (Figure 3). Two important features result from this ability. The first concerns the form of the VM itself: the VM environment may be different from that of the host hardware (but is usually in the host's family). Therefore, the host computer may support operating systems not specifically designed for it. This feature is most often exploited to provide backup and hardware changeover support (see Section V).

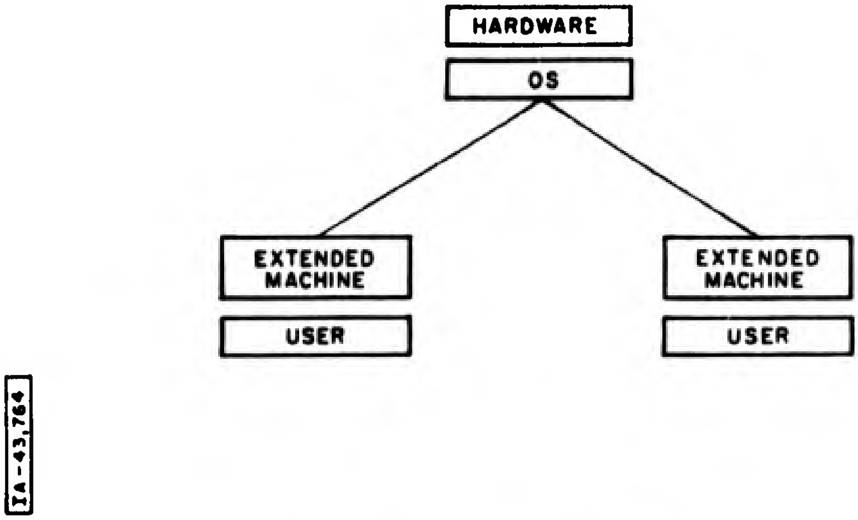


Figure 1 OS ORGANIZATION

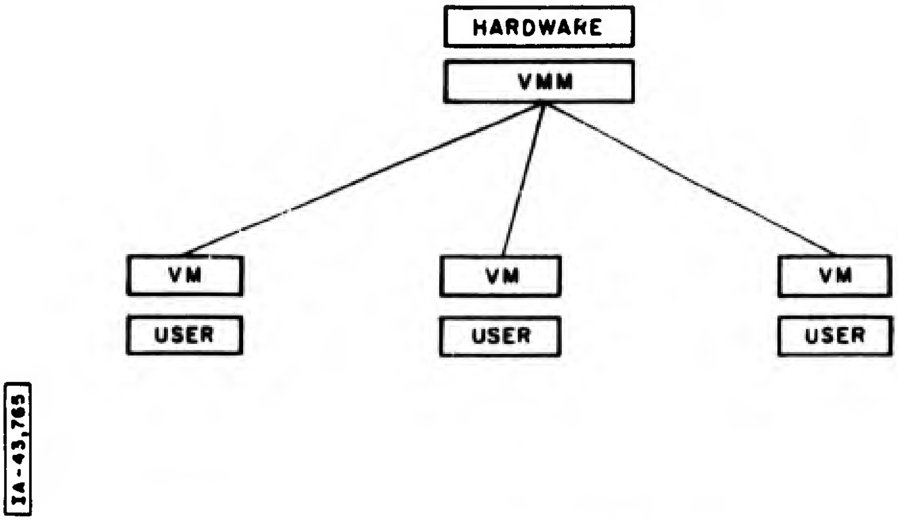
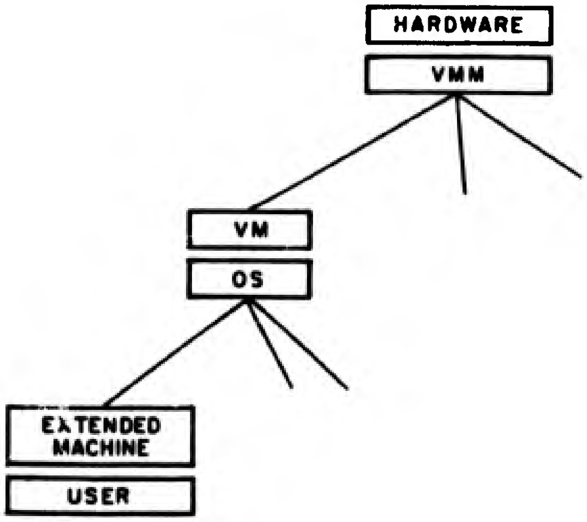


Figure 2 VMM ORGANIZATION



IA-43,766

Figure 3 TYPICAL VCS ORGANIZATION

The second feature concerns the multiplicity of the VM's; since several VM's may be supported concurrently, it is possible to distribute users onto different VM's. For instance, it would be possible to do test and development work on new operating systems without disturbing production runs on a different VM. In addition, the overall reliability of the facility is increased since user errors that "blow the operating system" only effect other users on the same VM and not the entire community.

VMM OPERATION

The method by which a VMM uses its control of hardware resources to support VM environments closely resembles OS multiprogramming. That is, the VM user (usually an OS) is given control of the hardware and allowed to execute normally until an instruction is attempted which requests resources regulated by the VMM. At this point a fault occurs and control is passed to the VMM for interpretation of the request.* After action on the request is taken, control is either returned to the same user at a point following the faulted instruction or returned to the user of another VM that was previously preempted. The cycle is then repeated.

In practice, the faulting mechanisms of the VMM and the OS are fundamentally different. The VM user, believing he is in full control of the computer, cannot be aware of the structure below him and any faults must be transparent to the user. On the other hand, the extended machine user is much more aware of OS activity as illustrated by the fact that many "faults" to the OS are calls to system routines.

The interpretation of resource requests by the VMM can be viewed as a mapping procedure by which VM resources are mapped to real hardware resources. It is through this mapping, then, that the VMM isolates the VM's from each other.

In order to apply the above mapping, resource requests must be trapped and referred to the VMM. For memory and I/O device resources, faulting mechanisms are in general straightforward and already exist on most third generation hardware. However, because the illusion of sole possession of the processor is being maintained for the VM user, the actual state of the CPU must be hidden in those cases where it differs from the state of the VM CPU. This constraint, in particular, is rather severe and prevents many processors from being used in virtual computer systems without some hardware modifications.

*For some resources, notably main memory, checking of the first request is usually done by VMM software with subsequent checks by hardware.

SUMMARY AND REFERENCES

This section has given a general outline of VCS organization and indicated the importance of VMM resource control in the maintenance of the isolated VM environments. Much more complete discussions of virtual machine architectures and specific examples may be found in the references (1-7). Goldberg[7] proposes a decomposition of the resource map that may enable easier, more efficient virtualization in future hardware. In addition, Popek and Goldberg[5] have formalized the constraints alluded to at the end of the section.

SECTION III

MULTILEVEL VCS

INTRODUCTION

By dedicating individual VM's to users and data of a specific level, a VCS is capable of multilevel operation. This multilevel VCS would provide security of the same granularity as that provided by the methods mentioned in the introduction; that is, inter-level access is strictly prohibited through the complete isolation of the various levels. As with the other methods, the OS for a given level (now running on a VM) provides a degree of control over interactions among users and data of the same level. However, because of the general unreliability of OS access controls, the effect of these intra-level controls is minimal.

The VCS approach to multilevel service differs from the previous methods in that all levels are simultaneously resident in the system and the (inter-level) security is enforced by logical rather than physical controls. It is primarily from the first of these differences, simultaneous residency of all levels, that the advantages of the VCS approach are derived:

(1) the service is provided by one set of hardware;

(2) transition between levels is smoother and more efficient than was possible with job scheduling techniques (due to the lack of lengthy manual changeover procedures);

(3) essentially continuous service may be offered to users at all levels through the use of relatively frequent level changes (made possible by (2)).

On the other hand, most of the difficulties of the VCS approach are a result of the use of logical controls. Specific drawbacks are: the need to implement a VMM--the software which enforces the logical access controls; the operational overhead associated with the VMM; and, most importantly, the problem of software certification must now be dealt with.

SECURE VCS AND SECURE COMPUTER SYSTEMS

It should be noted that, for the purposes of this paper, secure multilevel service has meant processing with complete isolation of individual levels.* In the context of the multilevel VCS developed above, then, the security of the system is equivalent to the isolation of the virtual machines. Therefore, (inter-level) security is maintained by insuring proper access of VM's to the hardware resources.

At this point, a parallel may be made to OS security considerations. The comparison is possible since the OS, in its attempt to maintain security, must deal with the similar notions of users (or processes representing them) requesting access to protected objects such as data files, I/O devices, etc. Moreover, the VM's and resources fit easily into the subject-object abstractions used for modeling OS security related functions. Once this relationship has been established, then it is possible to exploit current developments in secure general purpose computer systems to facilitate the planning of a multilevel VCS.

CERTIFICATION

The issue of system certification is crucial if the VCS is to be adopted to multilevel use. The physical isolation techniques used by present multilevel installations are, in general, inefficient and make use of the facilities inconvenient. However, these means were adopted because of a need for highly reliable security controls which were found to be lacking in available operating systems. Therefore, if the VCS is to provide a degree of security comparable to that of the present methods, a systematic approach is needed which will produce sufficient proof that the system has the desired features and that they work correctly.

The tendency to underrate VCS certification because of the seemingly complete isolation of the environments must be avoided. Similarities between virtual machine monitors and operating systems indicate that they suffer from approximately the same weaknesses. In particular, possible avenues of approach to the compromise of a VCS (i.e., operation of a VM user outside of his environment) might include the exploitation of flaws in crash recovery modules, VMM modules which execute user code while in the privileged mode of operation, or modules not equipped to correctly process an unusual series of requests.

Certification of a VCS is undoubtedly considerably simplified by the straightforwardness of the access controls and the small size of the VMM (as compared to a typical OS). However, this step in VCS development cannot be ignored if the system is to be completely trustworthy.

*This definition is somewhat more restrictive than the standard military-governmental one since inter-level accesses are not allowed.

It should also be emphasized that the use of penetration teams to establish system security is just as unreliable for the VCS as it is for the operating system. In general, the inability of any number of "experts" to gain illegal access does not guarantee that the next penetrator will fail. Rather some form of positive evidence must be supplied by a certification effort.[8]

REFERENCE MONITOR

The secure computer systems concept of a reference monitor[8,9] is easily adaptable to the VCS. In general terms, a reference monitor is a hardware-software entity which acts as the arbiter of all access requests. In order to facilitate later certification of the access controls of a system, it has been established that this monitor must be:

(1) always invoked before a request is granted;

(2) isolated from other supervisory functions;

(3) small and therefore easily understandable.

These properties are advantageous for aesthetic reasons alone; however, more importantly, they restrict the logical access controls to a much more manageable size and level of complexity. These considerations are particularly critical because of the severe limitations of presently employed certification techniques.

In the case of a VCS, all resource access requests are processed by the VMM. Therefore, the security kernel or software part of the reference monitor may be identified with some subset of the VMM. In general, the design principles listed above are inherent to a much larger extent in VMM design than OS design. Specifically: by its nature the VMM (or hardware set by the VMM) always mediates resource requests; the VMM, unlike the OS, performs few supervisory functions not directly related to the processing of resource requests; VMM's tend to be small as a consequence of the simplicity of the functions they perform.

While the first principle is necessarily a part of any VMM design, the degree to which the latter two principles can be incorporated is largely a function of the hardware. Ideally, the hardware allows a straightforward virtualization with a VMM that is small and limited to the processing of resource requests. In those situations which closely approach this ideal case, it is reasonable to identify the security kernel with the entire VMM. However, most third generation hardware is not completely compatible with a straightforward virtualization and consequently software which is not directly related to the mediation of access requests must be placed in the VMM. This software, while necessary for the maintenance of a correct VM interface, does not impact on security and should therefore be excluded from the security kernel as recommended by the second design principle.

CERTIFICATION DETAILS

System certification may be divided into two separate areas:

(1) verification that the security kernel design (or formal specification) correctly represents the logical access rules of the system;

(2) establishing that implemented kernel software reflects this design.

To enable certification of the first type, a "levels of abstraction" approach has been suggested for use in secure computer system work.[9] In this approach, a model of the reference monitor forms each level of abstraction with an abstract model of the security functions of the reference monitor forming the highest level. This upper level is hardware independent and deals only with the subject and object abstractions and the abstract rules of access imposed on them. Lower levels, then, are derived from higher ones in such a way that a lower level embodies the properties of its predecessor but with a greater degree of detail. The lowest level in the series most closely approaches an implementable design for the particular hardware to be used.

The first part of the certification process is now reduced to proving that the abstract model preserves security and a series of verifications that each level correctly represents its predecessor. Therefore, the overall process has been broken down into a series of more manageable sub-tasks. Moreover, in the case of the VCS reference monitor, the simplicity of the access controls and the functions performed by the VMM indicates that this part of the certification would be relatively straightforward as compared to the corresponding procedure for the secure OS reference monitor.

Other certification methods are available; in fact, the levels of abstraction approach as presented is not currently being considered for use in OS kernel certification (at least as far as the ESD/MITRE effort is concerned). An enumeration or evaluation of these methods is not particularly useful at this point since all such techniques are adaptable to VCS certification. However, it is important to realize that some form of well structured approach to VCS design is necessary. The relative simplicity of the VCS will only influence the quantity of possible errors and cannot significantly contribute to the goal of eliminating them entirely.

The second part of the overall system certification is subject to the same effect and, consequently, a more structured approach to kernel implementation will also be necessary if the elimination of all coding errors (i.e., differences in kernel design and implementation) is to be achieved. Procedures for this part of the certification process have not yet been established for use in the secure computer system effort of the USAF/ESD and MITRE. However, current work centered around the use of structured programming, proof of correctness and related areas is directed at the development of specific certification techniques.

It should also be noted that the necessary generality of the techniques being developed means that they can be applied without modification to any certification involving the correspondence between software design and implementation. Consequently, secure OS results in this area can be applied directly to VCS certification.

CONCLUSION

The preceding section has described the concept of a secure multilevel virtual computer system. In addition, through a comparison to OS security problems, an attempt has been made to emphasize the importance of an early and systematic treatment of security in the development of a VCS. Finally, a more specific approach to security has been derived from previous and ongoing work in secure computer systems. In particular, the concepts of reference monitor, security kernel and system certification have been applied to the VCS.

SECTION IV

APPLICATION CONSIDERATIONS

INTRODUCTION

The operational requirements of a particular installation, along with the properties of the host hardware, are intimately connected to the appropriateness of a multilevel VCS at that installation. Among the issues and tradeoffs in this area are:

- adaptability of hardware to VCS;
- granularity of security provided;
- means of deriving VM resources.

HARDWARE ADAPTABILITY

As Section II indicated, not all current systems have hardware that is amenable to a VMM. Fortunately, several widely used families of systems have members which, either directly or with some hardware modifications, are capable of providing a hardware base for a VCS.[2] In addition, current trends towards virtual storage and more reliable access controls indicate that VCS-supporting hardware may be much more common in the future. However, aside from the issue of VMM existence, the properties of the host hardware also strongly influence the efficiency of the final VMM. Considerations of this type are present in the administration of all hardware resources, but the problems are especially acute with the processor resource. The complex nature of this resource requires that some of its elements be completely or partially hidden from the VM user while others (such as the arithmetic and indexing registers, for example) are completely open. These hidden elements depend on the specific hardware being employed but may include indicator registers, interrupt masks, relocation-bounds registers or any other system information that might not reflect the state of the currently running virtual machine.

The general method of concealing the true hardware state of the system (indicated in Section II) is the interpretation by the VMM of all instructions which might reference hidden elements of the system. Interpretations of this type can be frequent and/or complex depending on the hardware base. Therefore, the properties of the processor can be seen to strongly influence VMM efficiency.

In one of the more extreme cases, the inability of the processor to correctly trap master mode instructions which manipulate or access hidden areas forces the interpretation and/or simulation of all instructions executed while the VM is operating in master (supervisory) mode. VMMs which maintain processor control in this manner are called hybrid virtual machine monitors[2] and, in general, produce VCS's which make poor use of the processor resource.

Typically, a hybrid monitor which took 20 machine instructions to interpret an average processor instruction and which supported an OS running 15% of its time in master mode, would require an average of 20 x (.15) + 1 x (.85) or 3.85 instructions for every instruction executed on the VM. That is, the real CPU usage for every job would be increased almost by a factor of 4 from interpretation alone! Of course, the hybrid VMM is a rather extreme solution; more commonly the VMM only interprets a few instructions and, ideally, only those instructions which actually do access hidden processor elements. In the latter cases, interpretation overhead can be small or even negligible.

Two other important considerations in the area of hardware adaptability should be briefly mentioned. The first concerns the direction of adaptation, that is, whether hardware modification can be used effectively to simplify virtualization or if additional software is the more appropriate way to go. The former would clearly produce a more efficient VCS but is not possible if off-the-shelf hardware is to be used or if useful changes would be too complex. Specific tradeoffs are, of course, determined by the host hardware.

The second additional consideration is related to the VM interface produced by the VMM. In general, the interface of any VM will differ in minor ways from the hardware interface of the "target" hardware. The most noticeable differences are related to timing considerations such as peripheral device delays or the calendar clock but minor variations in other areas are common. Tradeoffs arise because, in many cases, choices are present in the VMM design which may lead to a more efficient system whose virtual machines resemble the target hardware to a lesser extent than theoretically possible. The specific issues, in this case, are dependent on the host hardware, the target hardware and, to a large extent, on the peculiarities of the applications software.

GRANULARITY OF SECURITY

The proposed multilevel VCS, like present multilevel facilities that employ procedural job-scheduling techniques, is only capable of enforcing inter-level security. Within each level of the VCS (that is, on each VM) the users can only depend on that security provided for them by the resident operating system software. However, as Section III indicated, the protection provided by presently existing operacing systems is weak and easily breached by a penetration team.

Therefore, if the final system is to serve a fairly heterogeneous user community, a large number of levels may be required. The support of many levels, however, means an equal number of virtual machines and the accompanying inefficiencies due to:

- the large amount of multiprogramming done by the VMM;

- the amount of redundant supervisory code and software being executed and maintained in each VM environment.*

Because of these problems, the VCS may not provide a practical approach to some applications which require the isolation of many small groups of users.

Fortunately, most applications require few levels and consequently this type of overhead is minimized. The typical industrial installation might need only two levels - accounting and engineering. In terms of the military-government installation, levels could be supported which correspond to a subset of the levels in the standard classification system (unclassified, confidential, secret, and top secret) with a few additional levels for special access categories. In general, any <u>secure</u> multilevel facility today is restricted to a small (and therefore acceptable) number of levels by virtue of the inefficiencies inherent in the physical isolation approach.

MEANS OF DERIVING VM RESOURCES

In general, it is not practical to duplicate hardware resources in sufficient quantity to allow each VM to be permanently assigned its own share. It is therefore necessary to have the VMM control the sharing of these resources among the VM's. For many resources three methods of achieving this sharing are possible, with tradeoffs at a given installation being determined by the final application and available hardware. They are:

*This problem could be considerably reduced in future virtual computer systems provided the VM resident operating systems are identical and can be implemented as pure procedures. The amount of redundant code could be further decreased if a unique copy of the application software, system libraries, etc., for all operating systems is placed in "virtual" read-only memore (i.e., can not be set by a VM user).

- logically partitioning hardware resources - This method is most often applied to core, disk, and drum storage. Problems with the latter two arise from self-modifying channel programs -- resource requests that are altered after the request has been initiated. The existence of such requests indicates that there is inadequate hardware control of the partition and, consequently, extensive checking and/or simulation by the VMM is necessary to enforce the partitioning.

- allowing resources to be reassigned to different VM's - Use of this method for most resources is possible provided all information concerning the previous VM utilizing the resource is destroyed. This restriction is necessary to insure the isolation of the VM's. In addition, there is a more complex problem of insuring that no compromise can occur in the external (physical) computer environment of the computer room. At this time, the solution to the second problem appears to depend on providing a proper interface between the VMM and the operator to insure that no compromise can take place due to the improper marking of physical output.

- virtual resources - The substitution of plentiful resources for those in demand is common in virtual computer systems and is the underlying principle of virtual storage schemes in contemporary operating systems. In the VCS this substitution is possible because the VMM mediates all resource requests. Strictly speaking, this method is not distinct from the previous two since it is not a means of sharing a basic hardware resource, but rather a means of mapping it into a different resource. Consequently, provided the basic resource must be shared, the problems associated with virtual resources will predominantly depend on which one of the first two means of sharing is adopted.

It should be noted that resource sharing in an actual implementation might involve several of the above methods in combination; if so, there will be an accompanying compounding of the problems involved. As an example of such a situation consider a VCS that (like most today) 1) partitions core for its VM memories and 2) must support more VM's than can be resident in core at any given time. In this case both partitioning and reassignment techniques must be employed in order to share the core resource.

CONCLUSION

This section has attempted to give a brief outline of the tradeoffs and options involved in consideration of a VCS approach to multilevel security at a given facility. It should be apparent that design of a VMM is extremely sensitive to the properties of the hardware and in particular the processor. Consequently, matters of VMM design and efficiency must be considered on a case by case basis. Other factors contributing to the efficiency (or inefficiency) of the VCS are application related and include: the number of VM's to be supported; the size of software and data bases to be supported on each VM and; the adaptability of the software to a slightly different interface.

SECTION V

VCS APPLICATIONS

INTRODUCTION

There are essentially two approaches to providing secure multilevel service.[10] These methods can be distinguished by the objects which are protected in the interest of maintaining security; virtual computer systems, the approach discussed in this paper, relies on resource control to preserve security while the second method is based on the control of access to logical information objects.

Secure data descriptor based systems, i.e., those which employ the latter method, can be easily modified to provide multilevel service similar to that provided by a VCS. Specifically, all data descriptors and user identifications can be appended with level information and the access control mechanism can be modified to consider this information when granting access. In this way, users of one level can be completely excluded from data of any other level.

However, by more fully utilizing the generality of the descriptor based system, an improved type of "multilevel"* service can be offered in which various forms of controlled inter-level file accesses could be permitted. These expanded multilevel services are extremely useful since they fit many systems of information handling procedures that are not easily dealt with (or impossible) using the VCS approach. A noteworthy example of such a system is a standard military system which enforces need to know as well as classification and category restrictions. In this case, personnel and data of a given classification and category form a logical level and a partial ordering is imposed on the set of levels. Inter-level access is regulated by rules involving the partial ordering and, in addition, user access to individual data files may be regulated by designated "owners" of the data.

*It should be noted that systems which allow inter-level accesses require a more flexible definition of multilevel and security than has been used here. A specific example may be found in the mathematical model of Bell and LaPadula.[11]

IMMEDIATE USES OF A MULTILEVEL VCS

The multilevel VCS, in spite of its inability to cope effectively with the more general systems of access control just discussed, is a useful approach to providing multilevel service. Applications of the multilevel VCS include:

(1) interim solution to providing continuous multilevel service;

(2) applications requiring only isolated levels;

(3) transition aid to secure (descriptor based) computer systems.

Interim Solution

The VCS is capable of a faster switching rate between levels than is possible with the cumbersome manual changeovers of job scheduling; indeed, if transition overhead is sufficiently small, high switching rates may be used which provide essentially continuous service to all levels. Moreover, the relative simplicity of VCS access rules as compared to those of the secure OS (which must allow for the controlled sharing of information) means that implementation of the VCS reference monitor would probably be considerably simplified. Therefore, given the availability of hardware capable of supporting a VMM, the development of a multilevel VCS may provide an approach to providing interim continuous, multilevel service. In this case, the coarser granularity of the VCS security is advantageous since it enables more straightforward implementation of a secure system.

A related configuration, the Job Stream Separator,[12, 13] embodies VCS concepts without restricting the system only to hardware which can support a VMM. This versatility allows almost any computer to be retrofitted with security controls. Normal problems associated with virtualization are sidestepped by placing the VMM in a minicomputer and allowing each VM in its turn to control the entire main processor and memory (see Figure 4). Control of the peripheral devices is maintained by the minicomputer (VMM) either by placing switches controlled by the minicomputer on the data paths between the peripheral devices and the main computer or by having the devices attached directly to the minicomputer to facilitate dynamic reconfiguration.

Disadvantages of the minicomputer approach to multilevel processing are for the most part related to the time involved in changing levels. Specifically, the time to stop or wait for the completion of all current I/O before a level change is considerable. Moreover, the entire contents of core (or a large part of it) needs to be replaced at every change of levels. This overhead is significant and necessarily impacts unfavorably on the rate of level changes and the related issue of continuous on-line service to several levels.

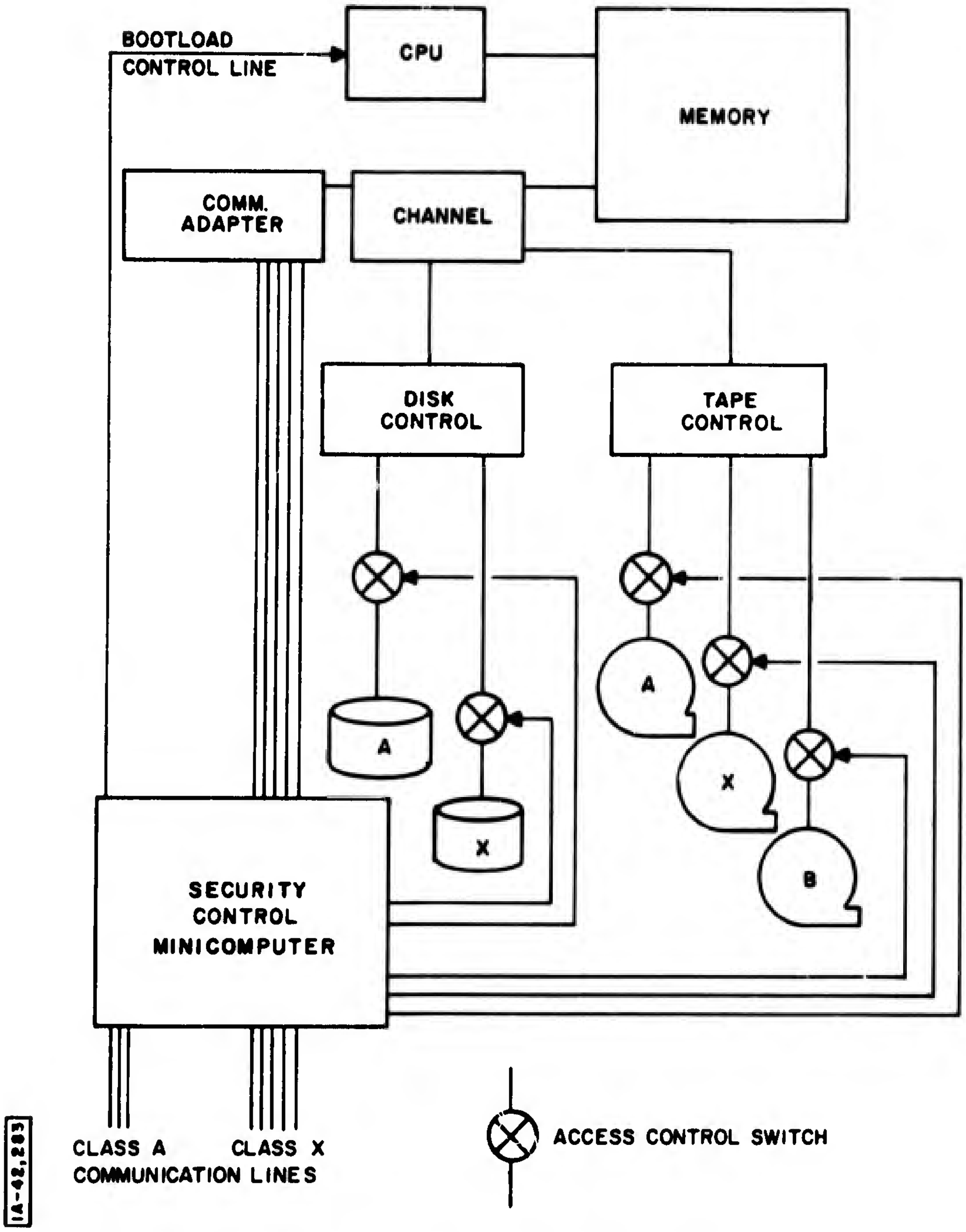


Figure 4 JOB STREAM SEPARATOR

Applications Needing Only Isolated Levels

The coarser granularity of VCS also means that less related system overhead is needed. Consequently, if the overhead associated with virtualization is relatively small, the VCS is an efficient alternative to the secure general purpose operating system in those applications which do not require any inter-level access.

In particular, the VCS is a useful tool for consolidating the activities of several facilities under one computer while still retaining their mutual isolation. In this application, the VCS is particularly useful because of the VMM ability to concurrently support VM environments resembling different sets of hardware (generally members of the same computer family as the host hardware). If a similar unification were attempted with a secure OS responsible for the isolation of these facilities, troublesome software changes might be necessary at some or all of the facilities to insure compatibility with the operating system software of the new system.

Associated with the previous application is the concept of a VCS backup support for facilities whose hardware environment can be duplicated by a VM. The VCS is suited to this application because of its ability to mimic diverse hardware environments simultaneously. As before, the mutual isolation of the different facilities temporarily using the same hardware is insured by the certified reference monitor within the VMM.

Transition

The VCS also has application in aiding the transition to a secure computer system (OS). Specifically, if hardware is available that can allow a VMM to support VM's resembling both the original hardware and the hardware environment required by the secure system, then the organization exemplified in Figure 5 could be employed to provide service to users of the old system and personnel requiring the new system interface. Therefore, production runs using the old system and software could proceed while test and development work is being done on the new secure system and its software.

The isolation of the VM environments in this case need not be related to security if the system is run at one level. However, if multilevel job streams are being run or if several VM environments resembling the original hardware are being used to provide multilevel service then certification of the VMM reference monitor would be necessary.

As illustrated by Figure 6, transition support may also be provided through the use of the so called Type II VMM[3] - a VMM

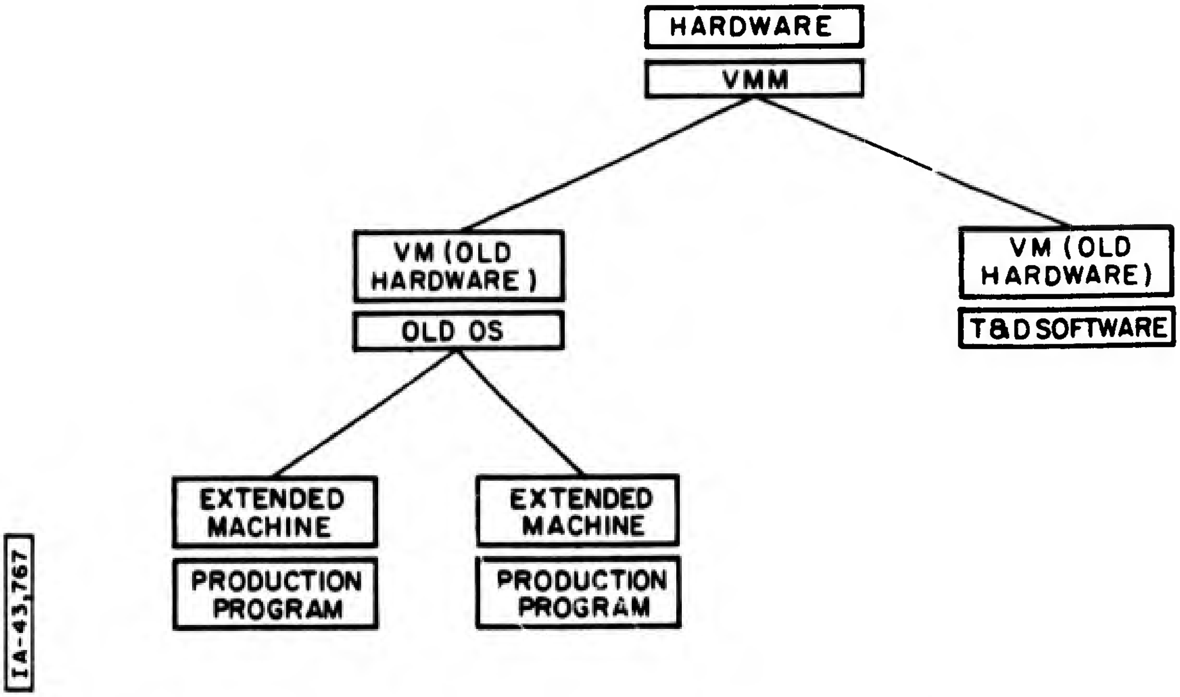
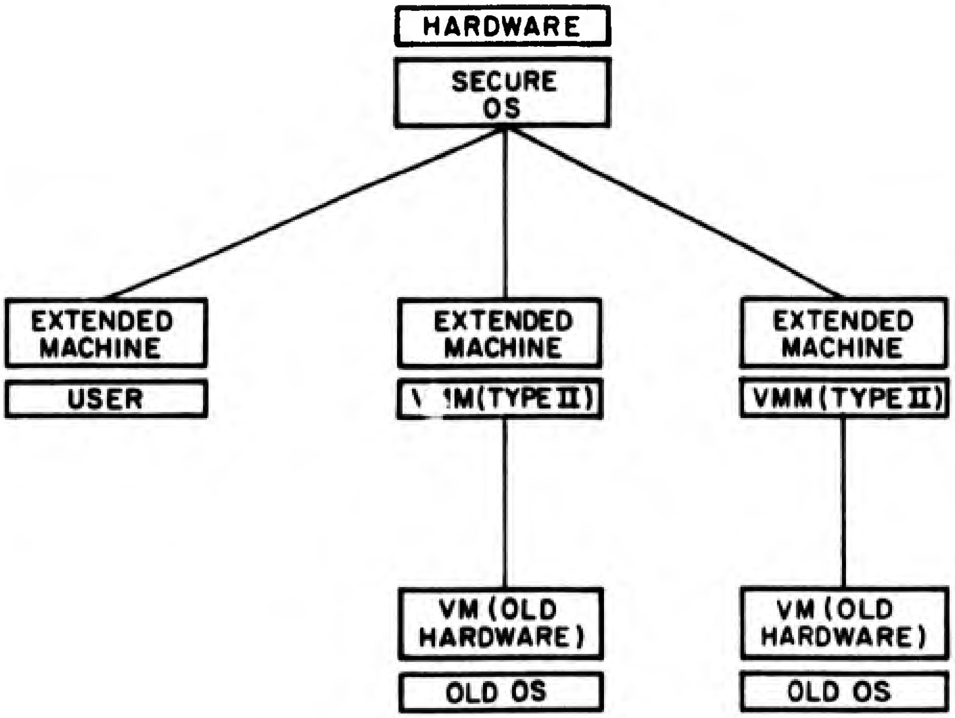


Figure 5 TRANSITION SUPPORT CONFIGURATION



IA-43,768

Figure 6 VCS EMPLOYING TYPE II VMM

running on the extended machine interface of an OS rather than a hardware interface. One advantage of this configuration is the fact that extended machines provide powerful tools that make VMM implementation easier. In addition, security considerations can be eliminated from VMM design since security can be guaranteed by the access controls of the secure operating system which supports the VMM.

On the other hand, the Type II VMM suffers the same weakness as the standard VMM in that its existence depends on properties of the host hardware that are not always present.[5] Moreover, the overhead associated with such a system is much higher than the usual VCS configuration because three rather than two layers of supervisory code are necessary to support the user.

The Type II VMM offers a high utility in situations where large and/or complex applications have already been implemented for the old hardware (and system) and would require reimplementation if they were to be run on the new hardware. With the capabilities provided by this VMM the need for reimplementation would not be immediate; it could even be avoided if the extra overhead were considered acceptable.

It should be noted, however, that the "transition support" provided by this configuration can only be made available after an operating system (and VMM) becomes operational on the new hardware. Consequently, only software conversion aid is provided to the user, not interim secure service.

THE FUTURE ROLE OF THE VCS

Drawbacks of the VCS approach to providing a general purpose multilevel computer utility can be divided into two classes: inefficiencies related to the inappropriateness of the hardware for virtualization; and inefficiencies and inflexibilities that are inherent in the VCS. Specific examples of the latter are the wasteful duplication of system and applications code in the VM environments (Section IV) and the lack of any type of information sharing between virtual machines (Section V).

Problems of the first type should largely disappear if newer hardware is designed with virtualization in mind. However, the second class of problems is strongly tied to the definition of a VCS since it is concerned with the non-intersection of the VM environments. If this latter set of problems can be dealt with, the VCS will provide a very attractive approach to providing multilevel security. An indication of the direction of possible extensions to VCS definitions or implementations can be seen from the following alternatives and options for future systems:

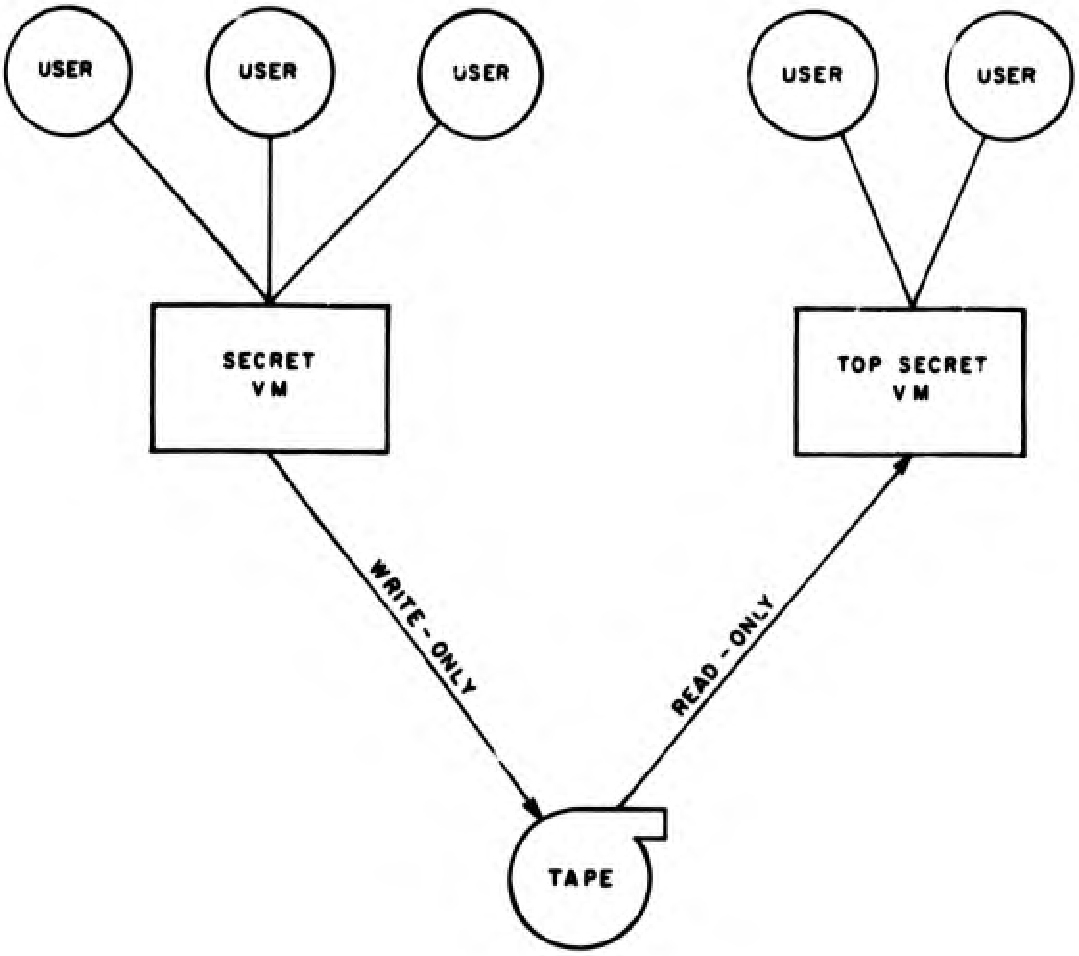
- Pure procedure OS code - The real memory resource can be more efficiently used by requiring that all VM resident operating systems be identical and be implemented as pure procedures. These requirements would enable the multiplexing of a unique copy of OS code among all virtual machines.

- Read only devices - The use of (possibly virtual) read-only devices would enable the storing of system libraries and other common files in only one physical location with the accompanying savings in real resources. Because these files would be read-only they could not serve as a data path between virtual machines.

- VM to VM communication mechanisms - In order to provide for the sharing of information between levels, virtual or real devices could be shared by virtual machines. Control over the sharing can be maintained to some extent by restricting the type of access (read or write) certain virtual machines have to the device. Another possibility that would allow for the information sharing between levels is the use of virtual inter-computer communication devices to accomplish the data transfer. In this case, control might be maintained by making the transfer unidirectional.

It should be kept in mind that any overlap between VM environments will probably require a reformulation of the concept of VCS security. In particular, the acceptability of inter-level communication would have to be evaluated in the light cf the formal data security requirements of the system. For instance, in the system illustrated in Figure 7, the VCS could enforce standard military access rules in regard to classification (i.e., the TS user may read S material but not the other way around). However, due to the lack of control on the part of the VM resident operating system, if several users are operating on each VM there is no means of enforcing need to know. This configuration may be reasonable at some installations but not discriminating enough in others.*

*Need to know could be enforced by a VCS if, fcr instance, each user is given his own VM. Devices could then be set up between VM's on the basis of the formal security rules.



IA-43,769

Figure 7 POSSIBLE INTER-LEVEL ACCESS MECHANISM

In any case, it should be realized that there are problems with data sharing between VM environments. Because the VMM only enforces a partitioning between the VM's and not within them, any data path out of an environment can potentially be used to compromise all information resident in the environment. Consequently, great care should be taken when considering the use of such data paths.

CONCLUSION

This section has attempted to present some of the specific applications in which a multilevel virtual computer system might be useful. In the area of general computer services, such systems as they exist today probably cannot compete with the more flexible descriptor based secure systems presently being developed. However, the use of virtual machines is a powerful tool in many specific situations and particularly in those areas which require interfaces different from that of the host hardware. Examples in this section have hopefully emphasized the present applicability of a multilevel VCS.

In addition, the second half of the section has identified some of the weaknesses inherent in the total isolation of VM environments. Several alternatives were suggested which illustrate the type of extensions that would be useful given a less restrictive definition of virtual machines. Specifically, these extensions provided for varying degrees of VM overlap to improve efficiency or convenience. In general, it is felt that future virtual computer systems can be an efficient, useful alternative to supplying a general purpose multilevel computer service provided:

(1) VCS hardware and software (including VM resident OS's) can be designed with virtualization and the efficiency of the virtual system as a primary consideration;

(2) there is some form of inter-level data sharing to enhance system flexibility;

(3) a systematic approach to VCS security is undertaken that employs the kernel, reference monitor and system certification concepts discussed in the paper.

REFERENCES

1. J. P. Buzen and U. O. Gagliardi, "The Evolution of Virtual Machine Architecture", Proceedings AFIPS National Computer Conference, 1973.

2. R. P. Goldberg, "Architectural Principles for Virtual Computer Systems", PhD. Thesis, Division of Engineering and Applied Physics, Harvard University, Cambridge, Massachusetts, 1972.

3. R. P. Goldberg, "Virtual Machines - Semantics and Examples", Proceedings IEEE International Computer Society Conference, Boston, Massachusetts, 1971.

4. R. P. Goldberg, "Architecture of Virtual Machines", Proceedings AFIPS National Computer Conference, 1973.

5. G. P. Popek, R. P. Goldberg, "Formal Requirements for Third Generation Virtual Architectures", Proceedings ACM SIGOPS Fourth Symposium on Operating Systems Principles, Yorktown Heights, N. Y., 1973.

6. R. P. Parmellee, T. I. Peterson, C. C. Tillman, D. J. Hatfield, "Virtual Storage and Virtual Machine Concepts", IBM System Journal, Vol. III, No. 2, 1972.

7. R. A. Meyer, L. H. Seawright, "A Virtual Machine Time-Sharing System", IBM System Journal, Vol. IX, No. 3, 1970.

8. J. P. Anderson, "Computer Security Technology Planning Study", James P. Anderson and Company, ESD-TR-73-51, Vol. I, Fort Washington, Pennsylvania.

9. R. R. Schell, P. J. Downey, G. J. Popek, "Preliminary Notes on the Design of Secure Military Computers", MCI-73-1, Deputy for Command and Management Systems (MCIT), Electronic Systems Division (AFSC).

10. J. P. Anderson, "Systems Architecture for Security and Protection", National Bureau of Standards Conference on Privacy and Security, 1974.

11. D. E. Bell, L. J. LaPadula, "Secure Computer Systems", The MITRE Corporation, ESD-TR-73-278, Vol. I-III, Bedford, Massachusetts, 1973.

REFERENCES (CONCLUDED)

12. S. B. Lipner, "A Minicomputer Security Control System", The MITRE Corporation, MTP-151, Bedford, Massachusetts, 1974.

13. R. Bisbey, G. J. Popek, "Encapsulation: An Approach to Operating System Security", USC/Information Science Institute, Marina del Ray, California, 1973.