PROCEEDINGS
of the

SEMINAR ON THE DOD COMPUTER SECURITY INITIATIVE PROGRAM

NATIONAL BUREAU OF STANDARDS
GAITHERSBURG, MARYLAND

JULY 17-18, 1979

TABLE OF CONTENTS

ABOUT THE SEMINAR

The objective of this seminar is to acquaint computer system developers and users with the status of the development of "trusted"* ADP systems within the Department of Defense and the current planning for the integrity evaluation of commercial implementations of these systems. The seminar will present an overview of a number of topics essential to the development of "trusted" ADP systems. Much of the material to be presented will be of a technical nature that is intended for computer system designers and software system engineers. However, the sophisticated computer user in the Federal government and in private industry should find the seminar useful in understanding security characteristics of future systems. This is the first in a series of technical seminars; future sessions will include detailed presentations on:

Security Kernel Design Experience

KSOS, KVM, SCOMP, Secure Unix Prototypes, MULTICS AIM

Specification and Verification Techniques

Secure System Applications

*A "trusted" ADP system is one which employs sufficient hardware and software integrity measures to allow its use for simultaneously processing multiple levels of classified and/or sensitive information.

ABOUT THE DOD COMPUTER SECURITY INITIATIVE

The Department of Defense (DoD) Computer Security Initiative was established in 1978 by the Assistant Secretary of Defense for Communications, Command, and Control and Intelligence to achieve the widespread availability of "trusted" ADP systems for use within the DoD. Widespread availability implies the use of commercially developed trusted ADP systems whenever possible. Recent DoD research activities are demonstrating that trusted ADP systems can be developed and successfully employed in sensitive information handling environments. In addition to these demonstration systems, a technically sound and consistent evaluation procedure must be established for determining the environments for which a particular trusted system is suitable.

The Computer Security Initiative is attempting to foster the development of trusted ADP systems through technology transfer efforts and to define reasonable ADP system evaluation procedures to be applied to both government and commercially developed trusted ADP systems. This seminar is the first in a series which constitute an essential element in the Initiative's Technology Transfer Program.

The Institute for Computer Sciences and Technology, through its Computer Security and Risk Management Standards program, seeks new technology to satisfy Federal ADP security requirements. The Institute then promulgates acceptable and cost effective technology in Federal Information Processing Standards and Guidelines. The Institute is pleased to assist the Department of Defense in transferring the interim results of its research being conducted under the Computer Security Initiative.

PROGRAM

SEMINAR ON
DEPARTMENT OF DEFENSE

COMPUTER SECURITY INITIATIVE

... to achieve the widespread availability
of trusted computer systems

July 17, 1979

8:30 am	Registration at National Bureau of Standards	
9:15	Opening Remarks	- James H. Burrows, Director Institute for Computer Sciences and Technology National Bureau of Standards
9:30	Keynote Address	- "Computer Security Requirements in the DoD" Honorable Gerald P. Dinneen Assistant Secretary of Defense for Communications, Command, Control and Intelligence
10:00		- "Computer Security Requirements Beyond the DoD" Dr. Willis Ware Rand Corporation
10:30	Coffee Break	
10:45		- DoD Computer Security Initiative Program Background and Perspective Stephen T. Walker Chairman, DoD Computer Security Technical Consortium
11:30		- Protection of Operating Systems Edmund Burke MITRE Corporation
1:00 pm	Lunch	
2:00		- Kernel Design Methodology LtCol Roger Schell, USAF Naval Post Graduate School
3:15	Break	
3:30		- Formal Specification and Verification Peter Tasker MITRE Corporation
4:30	Adjourn	

July 18, 1979

9:00 am		- Secure System Developments Kernelized Secure Operating System (KSOS) Dr. E. J. McCauley Ford Aerospace and Communications Corporation Kernelized VM-370 Operating System (KVM) Marvin Schaefer System Development Corporation
11:00	Coffee Break	
11:15		- Secure Communications Processor Matti Kert Honeywell Corporation
12:00		- Secure System Applications John P. L. Woodward MITRE Corporation
1:00 pm	Lunch	
2:00		- DoD Computer Security Initiative Stephen T. Walker
3:30	Adjourn	

KEYNOTE ADDRESS
on
COMPUTER SECURITY REQUIREMENTS IN THE DOD

by

DR. GERALD P. DINNEEN
Assistant Secretary of Defense (C^3I)

Welcome to the first seminar on the Department of Defense Computer Security Initiative. The goal of this initiative is (Slide available if you would like) to achieve the widespread availability of trusted computer systems. By "trusted", we mean systems with sufficient hardware and software integrity measures to allow their use for simultaneously processing multiple levels of classified or sensitive information. By "widespread" we imply the use of commercially developed trusted ADP systems whenever possible.

Let me begin with the understanding that today's computer systems in the DoD are secure in the physical, administrative, and procedural sense. We know how to treat computers like black boxes and lock them in physically secured facilities. However, with only a few exceptions, we are not able to rely on the integrity of the hardware and software components of our computers to properly isolate users from each others' data. We have therefore been forced to clear all users of a particular system to the same access level to prevent hardware or software failures from resulting in security violations.

As our computers become linked in worldwide data networks we can no longer afford to clear everyone to the highest level in the network. Even if we could, the "need to know" principle limits access to information to those who need it to perform their specific responsibilities.

"Information Exchange" is the key to success in any endeavor. In the DoD, and in particular in C^3I situations, the ability to accurately convey information is essential. Computers are becoming involved in every aspect of our information exchange activities. Until we can trust computers to accurately control access to critical information, our information exchange efforts will be seriously hindered. We have many ongoing programs which have strong requirements for trusted computer systems:

- WWMCCS has had a long stated requirement.
- The intelligence community has strong needs within its own community as well as in its interface to the rest of the national security community.
- Even our logistics and financial communities are faced with serious problems through the lack of trusted system developments.

Let me use as an example a hypothetical, but typical, automated text message handling problem. A nominal DoD message handling system might have twenty-five information sources and as many as two hundred distribution points. Some number of the information sources, say five, claim they cannot allow access to their information because there is not assurance that the information will not be sent to ten distribution points which are not cleared either for level or "need to know" access in all instances. To resolve this dilemma we have a number of possible solutions including:

1. Clear all distribution points to "too high" a level;

2. Set up multiple systems and somehow deal with the inherent problems such as maintaining multiple copies of data bases, plus the additional duplicate system expenses;

3. Deny some users access to data they need to do their jobs; or,

4. Build the message handling application on a trusted system base to maintain whatever access control processes are required.

Clearly, if possible, the last solution to build a trusted system is the most desirable and effective.

This is just one example of the problems we currently face, but I do not believe we should limit the computer security issue to only analyzing problems within existing programs. In a very real sense, the lack of trusted computer systems has limited to a degree the way we think about using computers. We have been so inhibited by our inability to trust computers that in many sensitive areas, our information exchange process has been warped. The existence of high integrity operating systems will create a dramatic shift in our ability to provide the right information to the right people at the right time.

In the next two days you will hear about technological developments which we feel will soon allow us to trust computer systems in many of our sensitive information handling environments. The developments you will hear about should not be interpreted as the ultimate answer but rather as a reasonable beginning. We feel confident that they are moving us on the path toward generally available trusted systems. I wish to emphasize that we do not today claim to have the solutions to all the problems. We do believe that the technology needed to build trusted systems is becoming available and we would like to encourage the development of trusted systems.

If we are to have widely available systems (that is, other than DoD developed special systems) we must involve the computer industry in this process. But before the industry will invest much effort in trusted systems, they must be convinced that such systems are reasonable for use on broader government and private sector sensitive information handling applications. Dr. Ware will address some of these beyond the DoD requirements in a few minutes. I want to emphasize that while we in the DoD may have a slight lead in understanding the nature of our sensitive information handling problems because of a long history of handling classified information, the general requirements of the rest of the Government and of the private sector are very similar to the kinds of problems we are trying to address. We hope that our efforts in this area may help to create the kind of trusted computer systems that can be used by anyone with sensitive information to process.

The program you will be hearing about in the next two days is <u>not</u> a Government R&D program involving grants to develop trusted computer systems. We believe that there will be sufficient market for trusted computer systems that the computer manufacturers will build high integrity trusted systems without the incentive for government development dollars. The Government R&D investment in this program is intended to demonstrate new approaches to solving the computer security problem but, beyond the initial demonstrations, our funds will be concentrated in trusted system applications.

Building computer hardware and software systems is a very complex process that the Government is no longer directly involved in except for special purpose systems that are unique to our needs. The large majority of our computer systems are purchased from the commercial market place. We realize that, if we are to achieve widespread availability of trusted systems, they must come from this same source. The DoD cannot afford, just for the sake of having trusted computer systems, to develop its own general purpose hardware and software systems. Further, we cannot afford to pay for special security related modifications to existing systems, with all the expensive long term maintenance implications. Nor do we believe that the manufacturers will spend their own IR&D funds to develop systems suitable for use just in the DoD market place.

Therefore, it has been an essential part of our computer security R&D program, to develop systems that are suitable for use in a wide variety of sensitive information handling environments (DoD, Federal Government, Private Sector), using software and hardware development techniques which are state-of-the-art and suitable for adoption by the computer manufacturers.

To the extent that we are successful, the manufacturers will find a much larger market for their trusted systems than just the DoD and the users of computers in sensitive information handling environments will find a significant improvement in the integrity of commercially available computer systems.

The technology for building computer hardware and software systems has always been an open, freely discussed and highly competitive field. The fact that we have made such significant advances in this field is due in large part to the openness which surrounds it. The basic technologies which we are employing to build and verify trusted computer systems (the technologies which you will hear about later today) are products of this open development environment. Yet as we employ these techniques on systems which will be used to protect sensitive information from improper disclosure or modification, we encounter a serious dilemma.

We know that no single security measure or combination of measures is 100% failure proof. Trusted computer systems while offering significant new capabilities to our computer usage also add an additional set of vulnerabilities to the overall security posture. We must recognize that, as with all other security mechanisms, there is always a potential for vulnerabilities with our hardware and software systems which we will have to protect by means of the physical and administrative measures that surround these systems.

We are going to have to draw a fine line between the openness with which we discuss computer systems development in general and the information restrictions we use to protect the potential vulnerabilities that may exist in the integrity measures of a particular system. We will have to develop procedures for protecting security relevant design and implementation details while not inhibiting general technological advances. I want to stress that this is not just a Department of Defense or U.S. Government problem but one which all of us face if we wish to develop or use high integrity systems. The solution to what information should be freely available and what should be restricted, and from whom it should be restricted, will not come easily and deserves the careful attention of all interested parties. I ask your help in arriving at a suitable solution.

In order to ensure that we can make the most effective use of trusted computer systems, we are attempting to establish an efficient and consistent evaluation process for determining the integrity of computer systems and the environments for which a particular system will be suitable. We hope to convey to you in the next two days some of the important technical elements that will influence this evaluation process. We are actively working toward establishing this evaluation process, though, I must emphasize, it may take some time to achieve our full objectives.

We feel it is essential to begin the dialogue on trusted computer systems with the computer system developers and major users immediately in order to ensure the earliest availability of these systems. This seminar is intended to initiate that dialogue. Our needs in this area are real and serious and we are eager to use trusted computer systems in existing and broad new applications as soon as they are available.

COMPUTER SECURITY IN CIVIL GOVERNMENT AND INDUSTRY

To begin, it is appropriate to share a bit of history. Figure 1 is the last of a series of slides that was first used at the (then) Spring Joint Computer Conference[1] of 1967. It was the first time that computer security as an explicit subject had been discussed for a technical audience. The slides gradually built up a resource-sharing, remote access computer system and at each step, indicated the vulnerabilities of each part. Figure 1 has been widely used by many authors to discuss the subject of computer security and to show system vulnerabilities.

Subsequently, the Advanced Research Projects Agency[2] organized a Task Force for the Defense Science Board to examine the subject; its report was published in 1970, the work having been done in 1968 and 1969.[3] Originally, the document was classified Confidential which limited its distribution largely to defense agencies and their contractors. In 1976, it was declassified and can now be freely distributed. Parenthetically, it might be noted that in spite of its classification, it was reviewed in an ACM publication--a unique event.

The insights and views of the Task Force are still valid; the report still is an outstanding exposition of the subject. Importantly, it introduced and established a consistent terminology. It also described in detail a scheme of access controls that can implement the information classification system of the defense establishment.

Turning now directly to the subject, let's ask when examining the computer security question in other than the Department of Defense: "Why do we want computer security safeguards?" There are three aspects of the answer. First, there must exist a body of information that for some reason is sensitive and must be protected. Second, there must be a threat against the information; for example, it might be pilfered, misused, or mistreated. Third, access to that information must be carefully controlled. These are the essential components of a requirement to provide computer security controls. In the DoD context, sensitivity of course relates to the formal classification scheme for defense and foreign policy information--collectively called national security

---

1. Now, the annual National Computer Conference sponsored by AFIPS.
2. Now, the Defense Advanced Research Projects Agency.
3. Security Controls for Computer Systems, 11 February 1970. Also published by the Rand Corporation as R-609 and reissued October 1979.

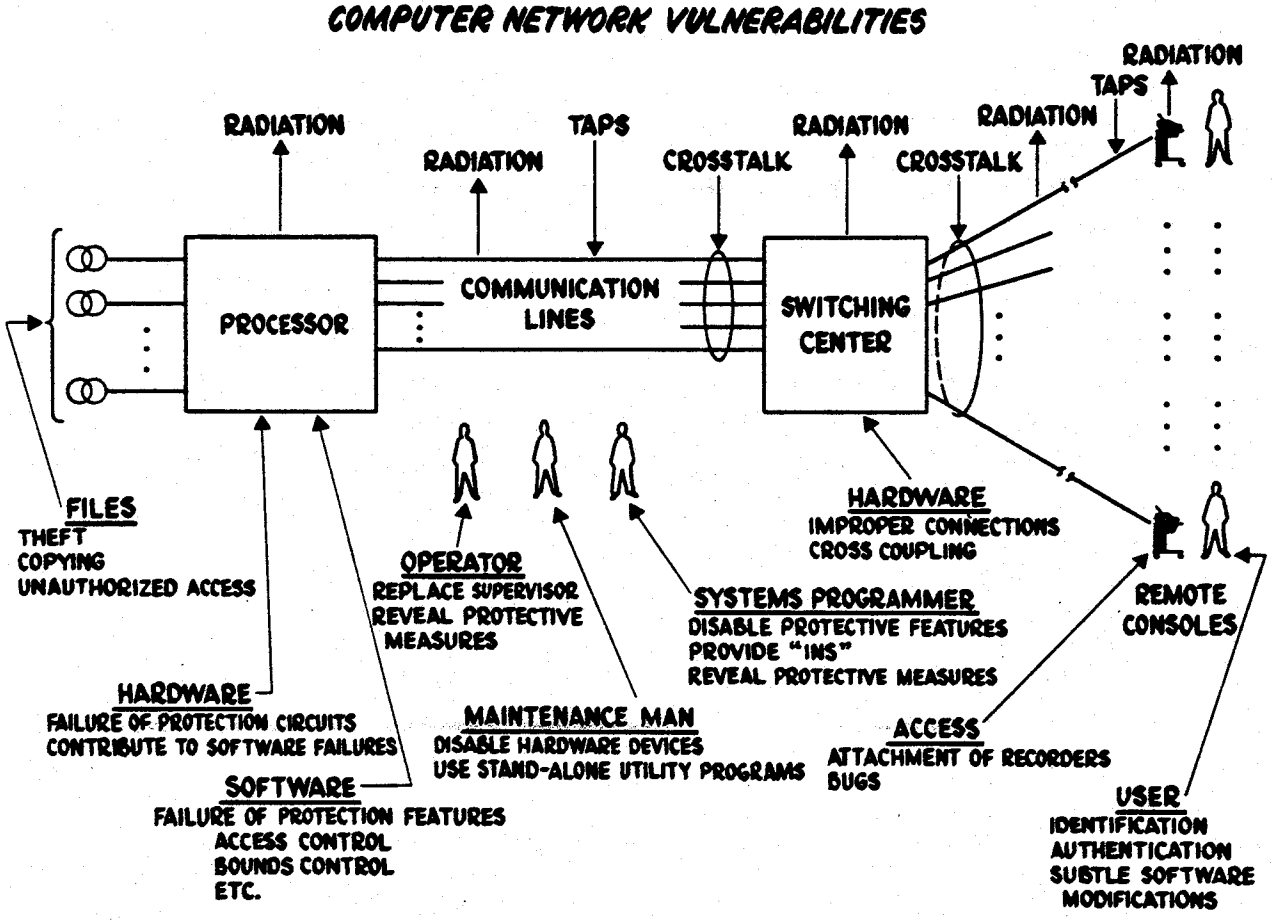

COMPUTER NETWORK VULNERABILITIES
RADIATION
TAPS
RADIATION
RADIATION
CROSSTALK
RADIATION
CROSSTALK
RADIATION
TAPS
PROCESSOR
COMMUNICATION LINES
SWITCHING CENTER
FILES
THEFT
COPYING
UNAUTHORIZED ACCESS
OPERATOR
REPLACE SUPERVISOR
REVEAL PROTECTIVE MEASURES
HARDWARE
IMPROPER CONNECTIONS
CROSS COUPLING
SYSTEMS PROGRAMMER
DISABLE PROTECTIVE FEATURES
PROVIDE "INS"
REVEAL PROTECTIVE MEASURES
REMOTE CONSOLES
HARDWARE
FAILURE OF PROTECTION CIRCUITS
CONTRIBUTE TO SOFTWARE FAILURES
MAINTENANCE MAN
DISABLE HARDWARE DEVICES
USE STAND-ALONE UTILITY PROGRAMS
ACCESS
ATTACHMENT OF RECORDERS
BUGS
SOFTWARE
FAILURE OF PROTECTION FEATURES
ACCESS CONTROL
BOUNDS CONTROL
ETC.
USER
IDENTIFICATION
AUTHENTICATION
SUBTLE SOFTWARE MODIFICATIONS

Figure 1

information. The threat against it is the established espionage effort of worldwide opponents, and finally, access control is strictly limited on a need-to-know basis.

Do similar sets of conditions exist elsewhere? Do they exist in civil government, business or industry? First, is there sensitivity of information in civil government? The answer is resoundingly yes. In the Federal Reserve system, there is an abundance of financial data pertinent to the country's welfare; in the Department of Agriculture, there is much information about commodity futures and prices; in the HEW, much information of a medical, health, and education nature; in Commerce, there is a huge data base that the Department of Census maintains; in the Department of Treasury, the IRS records; in the Social Security Administration, records of lifelong earnings. There are many kinds of information within civil government that are sensitive and must be protected, but is there a threat?

The answer again is yes. Buyers on the world market looking for good deals in wheat, for example, would be delighted to know what the Department of Agriculture thinks the future commodity market will be. Investigators and lawyers who are pursuing criminal cases or divorce cases or insurance claim cases are very interested in medical information and other data about people. Subversive elements that act to the disservice of the country would be delighted to subvert the financial data that the Federal Reserve maintains. Finally, there is the broad sensitivity of information gradually arising as privacy in recordkeeping comes into its own.[4]

Access control, the third component, is not well developed as an operational concept in civil government. To some extent it is recognized and practiced, but not as strongly as in defense. In civil government not everyone has access to everything, of course, but the formality of the matter is not strict in the same way as is in defense. In my view, yes, there is an argument for computer security safeguards in civil government.

What about business and industry? Sensitivity--yes indeed. The lifeblood of corporations is captured on magnetic discs and magnetic tapes--computer records. Such information is pertinent to competitive advantage, to corporate financial status, to pending sales, to details of orders, promotion

---

4. As, for example, stipulated by the Federal Privacy Act of 1974.

lists, and much else. It is all sensitive in one way or another.

Is there a threat? The question is harder to answer. If there is one, it is only dimly perceived and therefore one would have to conclude that it is largely latent. Some aspects of the threat are now appreciated, for example, the threat to corporate welfare from fires in computer rooms or from bombings. The threat from employees who exploit computer systems to rip the company off is beginning to be understood. Other aspects of the threat though are less obvious, and would have to be called latent. Fraud, embezzlement, theft of inventory, revelation of details about bids and proposals, and the like are obviously threats against sensitive information; regrettably the scope of the risk is weakly recognized by the corporate structure.

Access control is the third aspect. In the corporate structures of business and industry, the extent to which it exists is largely a result of historical practices rather than of a deliberately intended arrangement. Of course, only corporate officers see some things; and of course, only the corporate medical office sees certain things, but in the large, access control tends to be job-related. As for civil government, it does not have the formality and the strict need-to-know arrangement of defense.

In my view, civil government plus business and industry have exactly the same computer security needs as defense, but the need is not well-realized. Some details are obviously different but the broad principles are the same. When one has sensitive information in a computer, and when there is a threat against it, and when access to it must be controlled, then the three collectively point to a requirement for computer security safeguards.

The DoD happens to be ahead in calendar time for very good reason. It has several hundred years of military history that makes clear the threat against information. Moreover, it has decades of experience with the physical and personnel aspects of safeguards. The military establishment tends to have good institutional memory. The military schools, the writings of successful commanders, and the vivid experiences of a few wars collectively solidify the reality of the threat that the military establishment perceives against the information with

which it deals, and as well solidify the understanding that the information has to be protected.

Civil government, or business and industry does not have a corresponding attribute; each is a diffuse less tightly-knit community. In fact, the force of competition in business and the autonomy of various federal civil agencies act to keep it just that way--a loosely organized community. I suspect that a sense of reality toward information protection is gradually dawning, but there is about a decade difference between DoD awareness of the whole matter and other components of the country. Naturally there are isolated cases for which the time of difference does not exist, but generally it is there.

There is an important collateral comment. We all ought to be thankful for the Federal Privacy Act of 1974. While it did much for recordkeeping privacy, it also did much for computer security in the civil agencies. One particular provision of the Act stipulates that an agency must take reasonable precautions to safeguard the information which it holds. The consequence has been to overhaul casual computer operations and to remedy ill-advised information practices that had existed in civil agencies for a long time. The cause of computer security got an accidental but very helpful boost from the Privacy Act. For that we should be grateful.

From another point of view, computer security safeguards share an aspect with defense safeguards, as the latter are manifested in military forces and weapons. If one faces a disastrous event and overcomes it successfully, he clearly achieves an obvious explicit success and is acclaimed a hero. Computer security safeguards tend not to have such visibility. There may be times when it is clear that security safeguards have fended off an attempt to pirate sensitive information, or have fended off an attempted intrusive act; but, like defense forces, if computer security safeguards are working properly, nobody will ever know what undesirable things did not happen because of their presence. In a way they represent implicit successes because many attempted penetrations against a computer system will leave no trail of evidence that something was tried. For all of us who are convinced that computer security safeguards are important and must be provided, such a characteristic makes extra difficulty to justify and fund them.

There are some crucial differences between the computer security scene in defense and civil government. As you will hear throughout the symposium, DoD is facing the issue of how to certify that software is doing what it is supposed to be doing. Ideally, we could also certify that it does not do what it is not supposed to do. The DoD will have to make some arrangement to handle that matter; but it is not yet resolved. However that decision comes out, and however the DoD opts to handle the matter, the same solution may or may not be appropriate for civil government. The same question will have to be reexamined in the context of civil government.

Why might the DoD solution not be appropriate? Ideally, certified software will not be classified in the formal defense sense. Please note though that in the defense world, certified software operating systems will be used predominantly to protect official state secrets; maybe it will be decided that it cannot be unclassified. Everyone has hopes about how it will come out, but nobody can be sure just yet. Suppose it is necessary to classify the security controlling software, what will civil government do? Historically, it has not been involved with classified information; it does not understand what it means to clear people; it does not have experience dealing with secret information. Formal classification is a potpourri of things that civil government would prefer not to be involved with.

There is another difference of equal importance. The DoD is really a single agency; it presides over all military services and as such, the DoD has mechanisms for promulgating directives and policy which in turn guide and coordinate the services. There is no analogous unifying organizational structure in civil government that can cause all parties to march to the same drummer.

There is a third aspect. Generally speaking, the federal civil agencies really do not want to be in the computer business per se. Each is a consumer of computing power and each wants the information systems that computer systems provide; but really none wish to be in the computer business in the sense that we as technologists would use the phrase. Computer security is a complex matter--everyone is coming to understand that. Civil agencies need help and leadership; otherwise each will go its own way and there will be a repeat of the way agencies responded initially to the Privacy Act.

Each interpreted the Act its own way; there was no unifying discussion, no unifying leadership, no guidance, no guidelines. A similar thing cannot be allowed to happen in computer security which technically is much more complex than privacy.

At this point, it is useful to read two pertinent sections from the now unclassified DSB report. In the summary of the report, the Task Force reached certain conclusions and I call your attention to the first: "Providing satisfactory security controls in a computer system is in itself a system design problem. A combination of hardware, software, communication, physical, personnel, and administrative procedural safeguards is required for comprehensive security and in particular software safeguards alone are not sufficient." It was an important point to make at the time because people were arguing that the whole job could be done in software--so to speak, it's only a software problem. It was important for people to understand that computer security is a broad gauge system level problem. I observe in passing that the DoD may not yet have ssatisfactorily addressed the point. It is true that the present DoD initiative is addressing the operating system aspect; it is also true that other parts of the communication, physical, personnel, and administrative protections have been dealt with separately, but all pieces need to be assembled into a cohesive package.

The second quotation is an action item that the Task Force recommended to the Defense Science Board in February, 1970: "A technical agent must be identified to establish procedures and techniques for certifying security controlling systems especially the computer software portions and then for actually certifying such systems. The need for this agent is immediate but it will be difficult to create on short notice. System certification is a new technical area and substantial technical expertise in several disciplines is required. Two models come to mind for such an agent. The responsibility could be assigned to an existing agency of government, if it has the requisite skills (the Task Force suggested at that time in the defense context the NSA, DIA, and JTSA). Alternatively, an attractive idea is a multi-service agency operated and staffed by a contractor and created in the image of the Electromagnetic Compatibility Analysis Center." ECAC is a tri-service agency run and staffed by a contractor; it helps the military services deal with allocation and management of

electromagnetic spectrum space. It plays an essential roll as a tri-service entity but it is an image that is still a viable one for today's DoD need of certifying software.

Coming back to civil government, the image of a particular agency such as that of the DoD, or the image of a tri-service cooperative venture is probably viable. However, I know of one particular different precedent that is relevant and I submit that it is a useful model for us to think about.

The precedent that I would identify as pertinent is the role of the National Bureau of Standards Institute of Computer Science and Technology--ICST for short-- as an essential player in communication security for civil government. It was--and is--a pivotal player in the Digital Encryption Standard. What ICST did in getting the DES standard established was to provide for civil government an encryption methodology which prior to the time had been a capability available only in the defense establishment. If you will accept the phrase "technology transfer," here was an instance in which an operational capability through the effort of ICST was moved as a transfer of technology from its prior limited scope of applicability to all of government.

In handling the DES, ICST had the problem of examining the encryption algorithm and assuing its strength; it called on what I would consider the best resources of the governmment. It is a matter of public record that the National Security Agency assisted the National Bureau of Standards in testing and examining the algorithm and that it was satisfactory for encryption of communication traffic. I have to conclude that the NSA had the best resources of the government, or the arrangement would have not been made.

Now consider some questions pertinent to the role of ICST in computer security for civil government. Why should not the ICST take an analogous role in computer security as it did in DES? Why should it not finish the job that it really did start with DES? Why, under the auspices of the Institute, cannot the best resources within government, and if necessary from without government, be brought together to handle the remaining details of computer security safeguards? I would note particularly the question of how to certify secure software. Why should not the ICST create whatever additional Federal Information Processing Standards that are needed tc

finish the computer security job? Why should there not be FIPS that specify the performance requirements of a secure operating system plus the administrative, procedural and physical environments in which it has to be imbedded?

Why do I point this task toward the Institute? Simply, it is the only game in town. The ICST organization is the only one that I can see in civil government that has a chance of handling the problem. There is a legislative obligation to help; it has acquired the scope of technical expertise that I think is pertinent; it is the place in civil government where civil agencies should turn--have a right to turn--for leadership, expertise, guidance and whatever is needed not only in the whole matter of computer systems, but in particular the whole matter of computer security safeguards.

The omens are favorable. It has the mandate; it has the people; it has access to other parts of government: it has a new director who has long experience in defense and in government. To ICST and its director, I say: "You are up to it." To put it more pointedly, ICST must be up to it because there is no place else in civil government where the job can get done, and it does need to be done.

There are obviously a variety of sticky wickets ahead; and some are known. There are some obvious and genuine technical questions; there are some not so obvious jurisdictional ones; and there are the inevitable political ones. The job is not easy but on the other hand, things worth doing are never easy.

There is motivation to take advantage of this symposium to urge the NBS Institute of Computer Science and Technology to step out smartly on computer security, and I underline "smartly" because the need is developing fast. I could argue that it is regrettable for the Institute to be housed within an organization that happens to be called the National Bureau of Standards. The aura that goes with the concept of a standard is something that may take five, ten, fifteen, or a hundred years. Such is not the nature of business in the information and computer world. In a very real sense, ICST has to function on a time scale and act with a level of activity that is incommensurate with the usual concept of standards.

Notwithstanding, if the ICST will step out smartly, notice what can be achieved. First, government will have the needed new FIPS--Federal Information Processing Standards--that are needed to help civil agencies. The FIPS will have the force of a mandatory requirement; therefore, two important things will take place. From the Institute's activity, civil government will have the unifying leadership and unifying force that is needed. Moreover, vendors that wish to respond to government requests for business will have to provide hardware and software systems that contain appropriate security safeguards. In addition, they will have to provide the various technical, administrative, and educational materials for customer support. Thus, we will have also provided to business and industry the same security safeguards that are needed for its information protection.

I see this all as a fortuitous synergistic package of activities. I say again to the ICST: "Step out smartly, please." Civil government needs you; industry and business will receive enormously important collateral advantages.

DOD COMPUTER SECURITY INITIATIVE PROGRAM

BACKGROUND AND PERSPECTIVE

Stephen T. Walker
Chairman, DoD Computer Security
Technical Consortium

COMPUTER SECURITY INITIATIVE

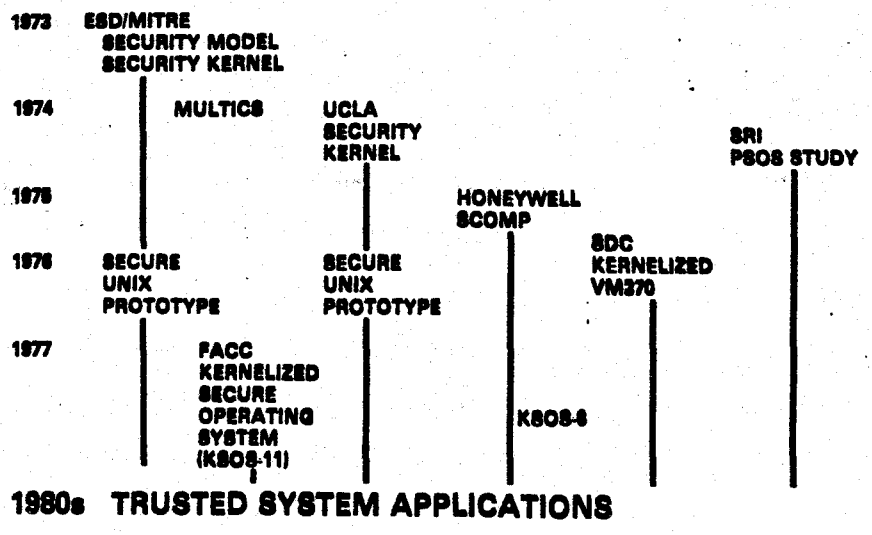
GOAL: WIDESPREAD AVAILABILITY OF "TRUSTED" ADP SYSTEMS

- **"TRUSTED" MEANS: SUFFICIENT HARDWARE AND SOFTWARE INTEGRITY TO SEPARATE AUTHORIZED USERS WITH DIFFERENT ACCESS PRIVILEGES**
- **WIDESPREAD AVAILABILITY MEANS: COMMERCIALLY AVAILABLE**

COMPUTER SECURITY EVOLUTION

1967	DEFENSE SCIENCE BOARD = WARE REPORT
LATE 60s - EARLY 70s	PENETRATION EFFORTS
1972	AF ESD PANEL = REFERENCE MONITOR CONCEPT

COMPUTER SECURITY EVOLUTION



KERNELIZED SECURE OPERATING SYSTEM (KSOS)
"SECURE UNIX"*

1976-1977 – UCLA AND MITRE SECURE UNIX PROTOTYPES

AUG 1977 – COMPETITIVE PROCUREMENT, TWO DESIGN PHASE CONTRACTS TRW, FORD AEROSPACE & COMMUNICATIONS CORP.

MAY 1978 – IMPLEMENTATION PHASE CONTRACT: FORD AEROSPACE

FEB 1980 – ALPHA TEST SITES

JUN 1980 – BETA TEST SITES

FALL 1980 – AVAILABLE AS SUPPORTED PRODUCT

*BELL SYSTEM TRADE/SERVICE MARK

KERNELIZED VM370

- TRUSTED VERSION OF IBM VM370 OPERATING SYSTEM
- GUARANTEE SEPARATION OF VIRTUAL MACHINES PROVIDED BY VM370
- 3 YEAR EFFORT, BEGUN IN MARCH 1976
- DEMONSTRATION - SUMMER 1979
- WORK PERFORMED BY SYSTEM DEVELOPMENT CORPORATION

Edmund Burke
MITRE Corporation

PROTECTION IN OPERATING SYSTEMS:

A TECHNICAL HISTORY

OUTLINE

- INTRODUCTION
- EARLY COMPUTER USE
- THIRD GENERATION SYSTEMS
- THIRD GENERATION PROBLEMS
- DEVELOPMENTS IN PROTECTION MECHANISMS

INTRODUCTION

- ACCESS CONTROL IS THE ISSUE
- POLICY DICTATES ACCESS CONTROL RULES
- MANDATORY POLICY
  - INFORMATION HAS CLASSIFICATION
  - PEOPLE HAVE CLEARANCES
  - CLEARANCES ≥ CLASSIFICATION
- DISCRETIONARY POLICY (NEED-TO-KNOW)

INTRODUCTION

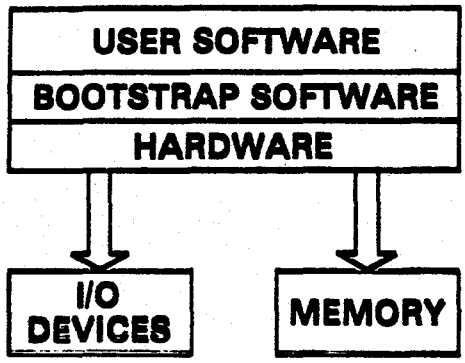
- TECHNIQUES EXIST FOR PEOPLE, PAPER WORLD
  - PHYSICAL
  - PERSONNEL
  - PROCEDURAL
- COMMUNICATIONS SECURITY EXISTS AND IS EVOLVING
- THE COMPUTER INTRODUCES A NEW DIMENSION
  - HARDWARE AND SOFTWARE CONTROLS

INTRODUCTION

- FIRST INTRODUCTION OF COMPUTERS POSED LITTLE PROBLEM
- COMPUTERS FIRST PROVIDED COMPUTATION
- LATER COMPUTERS DEALT WITH INFORMATION PROCESSING
- PROBLEMS AROSE WHEN RESOURCES WERE SHARED

FIRST GENERATION COMPUTERS

- FROM MID-40's TO CIRCA 1960
- HARDWARE AND SIMPLE BOOTSTRAP SOFTWARE

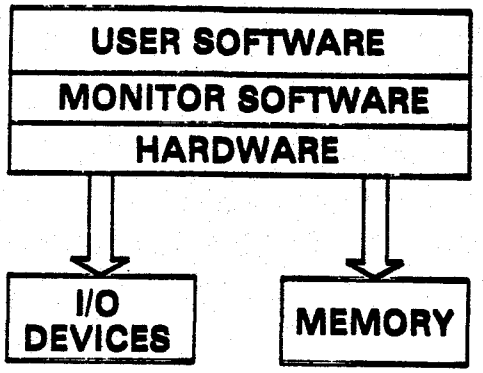


FIRST GENERATION SECURITY

- USER HAD ACCESS TO ALL PHYSICAL RESOURCES
- SYSTEM PROTECTED AT HIGHEST LEVEL
- MOST COMPUTATION DONE AT SINGLE LEVEL

SECOND GENERATION SYSTEMS

- CIRCA 1960 TO MID-60's
- MONITOR SOFTWARE



SECOND GENERATION SECURITY

- USER HAD ACCESS TO MOST RESOURCES
- SYSTEM, USERS AT HIGHEST LEVEL
- SOME MANUAL REVIEW FOR LOWER CLASSIFIED RUNS

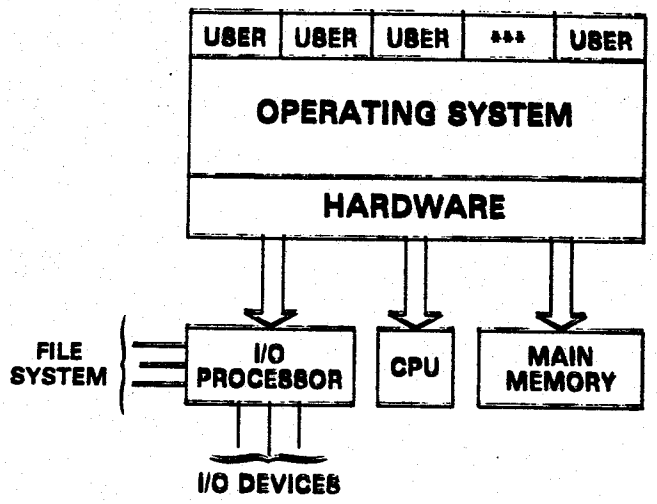
SECOND GENERATION SYSTEMS

- NOTION OF "LOGICAL" I/O DEVICES
- MEMORY SPACE OVERLAYED
- LIBRARY FUNCTIONS
- USER AT A TIME
- THE BEGINNINGS OF TIME SHARING/INFORMATION PROCESSING

THIRD GENERATION SYSTEMS

- SYSTEM COULD SUPPORT MANY USERS
  - NEED TO SHARE RESOURCES
- BOTH INFORMATION PROCESSING AND COMPUTATION
- WIDER RANGE OF COMPUTATIONAL SERVICES
  - LANGUAGES
  - DMS
  - OTHER FEATURES

THIRD GENERATION SYSTEMS



THIRD GENERATION SERVICES

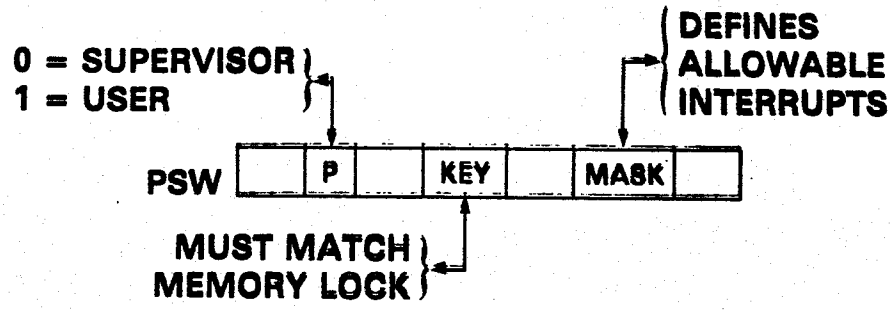
- RESOURCE ALLOCATION
  - CPU SCHEDULING
  - I/O DEVICE SCHEDULING
  - MEMORY SPACE
- FILE SYSTEM
  - VIRTUALIZATION OF FILE RESOURCES
- PROGRAM LIBRARY
  - LANGUAGES
  - SUBROUTINE PACKAGES

THIRD GENERATION PROTECTION

- HARDWARE MECHANISMS FOR RESOURCE SHARING
  - I/O & CPU USAGE CONTROLLED BY
    - PRIVILEGED INSTRUCTION
    - SUPERVISOR/USER DOMAIN
    - INTERRUPT MECHANISM
  - MEMORY PROTECTED BY KEYS

THIRD GENERATION PROTECTION

- HARDWARE PROTECTION CENTERS AROUND PROGRAMS
- KEY FEATURE IS PROGRAM STATUS WORD (PSW)



THIRD GENERATION PROTECTION

- ACCESS TO RESOURCE GOVERNED BY PSW
  - SUPERVISOR STATE ALLOWS
    - PRIVILEGED INSTRUCTIONS
    - USER STATE INTERRUPTS
    - I/O INTERRUPTS
  - BOTH STATES MUST HAVE KEYS

THIRD GENERATION PROTECTION

- SOFTWARE AUGMENTS HARDWARE
  - CHECKS LEGALITY OF SUPERVISOR CALLS
  - DOES MOST I/O PROCESSING
  - PROVIDES PRIMITIVE FILE PROTECTION (PASSWORDS)
  - AUDIT MECHANISM

THIRD GENERATION PROTECTION SUMMARY

- PROGRAM IS ACTIVE INFORMATION ACCESSOR
  MECHANISMS PROTECT PROGRAM ACCESS TO RESOURCES
- CPU, I/O, MEMORY, FILES ARE PROTECTED ENTITIES
- SOFTWARE FEATURES BOLSTER HARDWARE AND SOFTWARE MECHANISMS

THIRD GENERATION SECURITY

- USERS AND INFORMATION AT DIFFERENT LEVELS
- WOULD LIKE TO RELY ON HARDWARE AND SOFTWARE ACCESS CONTROLS
- FLAWS WERE FOUND
- HAD TO REVERT TO TRADITIONAL TECHNIQUES

THIRD GENERATION PROBLEMS

- HARDWARE & SOFTWARE FEATURES WERE NOT EFFECTIVE
  - O/S DESIGN CONCEPTS WERE EVOLVING
  - NO WELL CONCEIVED DESIGN STRATEGY WITH CENTRALIZED ACCESS CONTROL
  - SIZE LED TO COMPLEXITY WHICH LED TO BUGS
- THE PROGRAM WAS NOT THE RIGHT SURROGATE FOR THE USER

THIRD GENERATION PENETRATIONS

• OS/360	MITRE
• GCOS	GOVERNMENT
• OS/VS	SDC
• UNIVAC 1108	NRL

PENETRATION EXAMPLES

- OS/360
  SUPERVISOR CALLS NOT PROTECTED
- GCOS
  CHECKPOINT DUMPS NOT RESTRICTED
- OS/VS
  MISUSE OF I/O PROCESSOR

THIRD GENERATION ADP SECURITY TECHNIQUES

- SYSTEM HIGH OPERATION
  - CLEAR EVERYBODY, EVERYTHING
- PERIODS PROCESSING
  - PROCESS DIFFERENT LEVELS DURING DIFFERENT TIME PERIODS

OPERATING SYSTEM DEVELOPMENTS

- THE THRUST WAS TO GIVE USERS A MORE TRANSPARENT SET OF CONTROLS
- RESOURCES WERE VIRTUALIZED
- SHARING OF RESOURCES BECAME MORE EXPLICIT
- "THIRD AND A HALF" GENERATION – MULTICS, TENEX, UNIX

OPERATING SYSTEM DEVELOPMENTS

- RESOURCE VIRTUALIZATION
  - MEMORY – VIRTUAL MEMORY
  - CPU – TIME SHARING
  - I/O – SERVICE PROCEDURES AND FILE MANAGEMENT
- "STRONGER" HARDWARE MECHANISMS WERE DEVELOPED
- MORE COHERENT SOFTWARE STRUCTURE WAS IMPOSED

MULTICS HARDWARE

- MEMORY MAPPING FOR VIRTUAL MEMORY
  - EACH VIRTUAL SEGMENT HAS A DESCRIPTOR
  - ACCESS CONTROL BITS IN DESCRIPTOR
  - DESCRIPTOR CACHE FOR PERFORMANCE
- RINGS GENERALIZE SUPERVISOR/USER CONCEPT
  - SUPPORTS ARGUMENT VALIDATION
- PROCESS NOTION SUPPORTED IN HARDWARE
  - PROCESS CONTEXT DEFINED BY ADDRESS SPACE AND POINT OF EXECUTION

MULTICS SOFTWARE

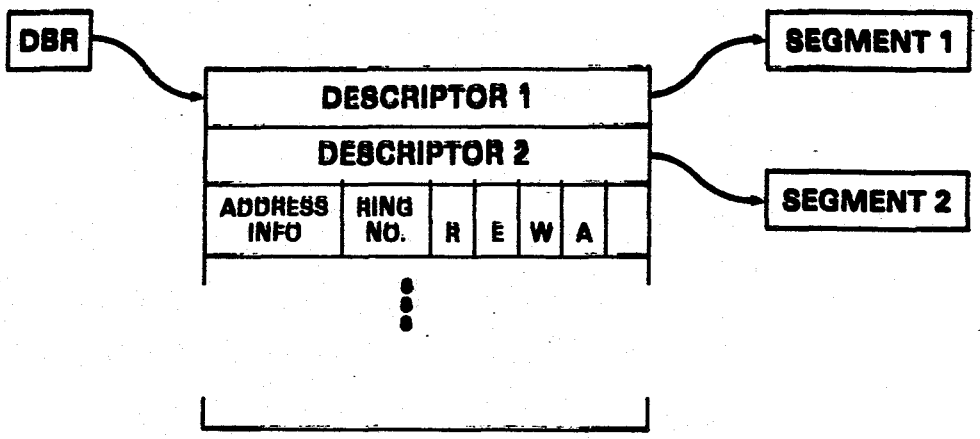
- MODULAR; WRITTEN IN PL/L, RENTRANT, SUPPORTS PROCESS-PER-USER CONCEPT
- CENTRAL ACCESS CONTROL POLICY (DISCRETIONARY)
- FILE SYSTEM INTEGRATED INTO I/O
- "HARD-CORE" O/S VS. OTHER USER SERVICES

SUMMARY

- O/S DESIGN EVOLVED (AND STILL EVOLVING)
- EFFECTIVE SHARED RESOURCES WAS THE GOAL
- ACCESS CONTROL A FACTOR IN RESOURCE SHARING
- O/S DEVELOPMENT NATURALLY ADDRESSING SECURITY ISSUES

PREVIEW

- KERNELS ARE SMALL, PRIMITIVE OPERATING SYSTEMS THAT
  - PROVIDE COMPLETE MEDIATION
  - ARE ISOLATED, AND
  - CAN BE VERIFIED TO OPERATE CORRECTLY


DBR
DESCRIPTOR 1
DESCRIPTOR 2
ADDRESS INFO
RING NO.
R
E
W
A
SEGMENT 1
SEGMENT 2

SECURITY KERNEL DESIGN METHODOLOGY

BY

ROGER R. SCHELL, LT. COL., USAF
ASSISTANT PROFESSOR OF COMPUTER SCIENCE
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

DEFINITION

"COMPACT SECURITY KERNEL OF THE OPERATING SYSTEM AND SUPPORTING HARDWARE —SUCH THAT AN ANTAGONIST COULD PROVIDE THE REMAINDER OF THE SYSTEM WITHOUT COMPROMISING THE PROTECTION"

— SCHELL 1972

OVERVIEW

- INTRODUCTION
- WHAT DOES "SECURE" MEAN?
  - REFERENCE MONITOR
  - SECURITY POLICY
- WHAT DOES A SECURITY KERNEL DO?
  - PROTECTION MODEL
  - INTERFACE PRIMITIVES
- HOW IS A SECURITY KERNEL IMPLEMENTED?
  - PROCESSES
  - SEGMENTATION
  - PROTECTION DOMAINS
- SUMMARY

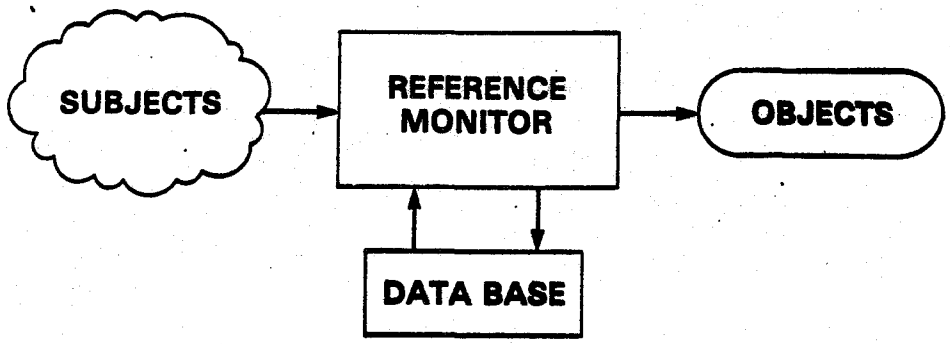
NATURE OF PROBLEM

- ENGINEERING ISSUES
  - MECHANISM COMPLEXITY
  - PROTECTION RULES
- FOCUS ON INTERNAL CONTROLS
  - OPERATING SYSTEM & HARDWARE
  - EXTERNAL CONTROLS ASSUMED
- DESIGN AS METHODICAL STEPS

REFERENCE MONITOR CONCEPT

- ABSTRACTION OF PROTECTION
  - KERNEL IMPLEMENTS
- COMPONENTS
  - SUBJECTS
  - OBJECTS
  - AUTHORIZATIONS
- TWO CLASSES OF FUNCTIONS
  - REFERENCE
  - AUTHORIZE
- FUNDAMENTALLY INTERPRETIVE

REFERENCE MONITOR CONCEPT

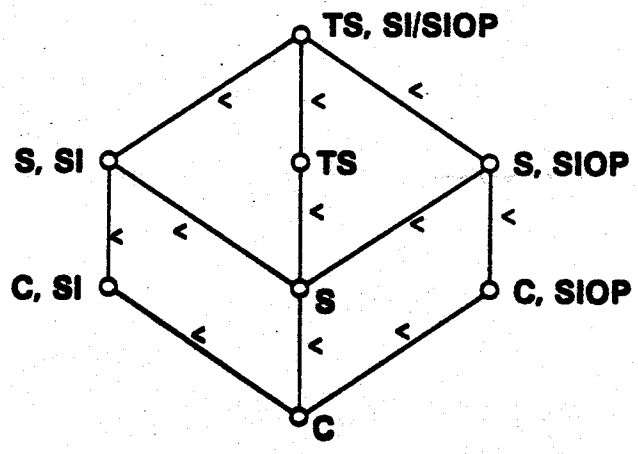


- IMPLEMENTATION CHARACTERISTICS
  - COMPLETENESS
  - ISOLATION
  - VERIFIABILITY

SECURITY POLICY

- DEFINED EXTERNALLY
  - LAWS, REGULATIONS, RULES
  - INDEPENDENT OF COMPUTER
- "SECURE" IS ONLY WITH RESPECT TO
- DICTATES BEHAVIOR, NOT FUNCTIONS
- TWO CLASSES FOR SECURITY KERNEL
  - NON-DISCRETIONARY
  - DISCRETIONARY

NON-DISCRETIONARY EXAMPLE



ACCESS CLASS RELATIONS

NON-DISCRETIONARY

- MANDATORY POLICY—MOST IMPORTANT
  - PERMISSIBLE ACCESS
  - EXTERNAL AUTHORIZATION
- UNIFORM LABELING VIEW
  - COMPARTMENTS
  - LEVELS
  - "LATICE POLICIES"
- SYSTEM (KERNEL) IS ENFORCER
- NOT COMMON OFFERING
  - HONEYWELL MULTICS "AIM" DEMONSTRATES

DISCRETIONARY

- FINER GRANULARITY—LESS IMPORTANT
  - WITHIN NON-DISCRETIONARY
  - NEED-TO-KNOW
  - MAY EXCLUDE FROM KERNEL
- USER (SUBJECT) IS ENFORCER
  - AUTHORIZES OTHERS
  - INTERNAL DISCRETION
- SYSTEM ADMINISTERS
- COMMON OFFERING
  - ACCESS LISTS, FILE PASSWORDS, ETC.

PROTECTION MODEL

- RULES MODEL KERNEL FUNCTIONS
  - REFERENCE MONITOR PARTICULARIZATION
  - BEHAVIOR PER POLICY
- STATE MODELS AUTHORIZATIONS
  - ACCESS CLASS LABELS
  - "ACCESS MATRIX"
  - SUBJECTS & OBJECTS

MODEL RULES

- NON-DISCRETIONARY REFERENCE
  - READ: LABEL (SUBJECT) > LABEL (OBJECT)
  - READ/WRITE: LABELS EQUAL
  - WRITE: LABEL (SUBJECT < LABEL (OBJECT)
- DISCRETIONARY REFERENCE
  - PER ACCESS MATRIX
- AUTHORIZE
  - SUBJECT CHANGES ACCESS MATRIX

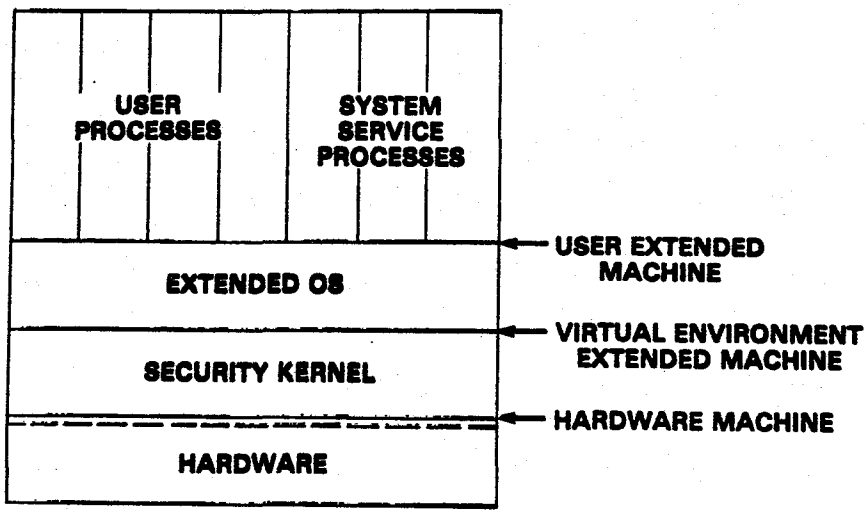
MODEL SIGNIFICANCE

- SELECT SECURE INITIAL STATE
- FOLLOW RULES FOR CHANGE
- ALL STATES ARE SECURE

SECURITY KERNEL PRIMITIVES

- SET OF "SUPERVISOR CALLS"
  - SPECIFIC MODEL INTERPRETATION
  - VIRTUALIZES RESOURCES
  - INVOKED BY OPERATING SYSTEM
- DATA BASES
  - REPRESENT MODEL STATE
- PERMIT ONLY AUTHORIZED REFERENCES
- TWO ELEMENTS
  - DISTRIBUTED KERNEL—IN EACH PROCESS
  - KERNEL PROCESSES

SECURITY KERNEL STRUCTURE



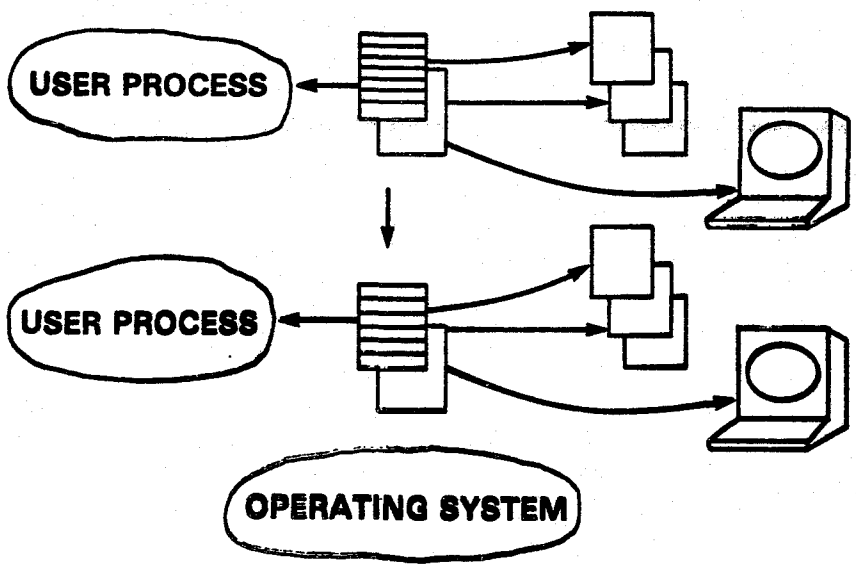
ILLUSTRATIVE PRIMITIVES

- CREATE/DELETE OBJECT
- GRANT/RECIND (DISCRETIONARY) ACCESS
- INITIATE/TERMINATE OBJECT ADDRESSABILITY
- SEND/RECEIVE INTERPROCESS MESSAGES

EXPLICIT PROCESSES

- REALIZE SUBJECT ABSTRACTION
- SECURITY ATTRIBUTES
  - NON-DISCRETIONARY LABEL
  - SUBJECT (USER SURROGATE) IDENTIFICATION
- SECURE SYNCHRONIZATION
- VIRTUALIZE CPU & I/O PROCESSORS
- TWO CHARACTERISTICS
  - EXECUTION POINT—PROCESSOR STATE
  - ADDRESS SPACE

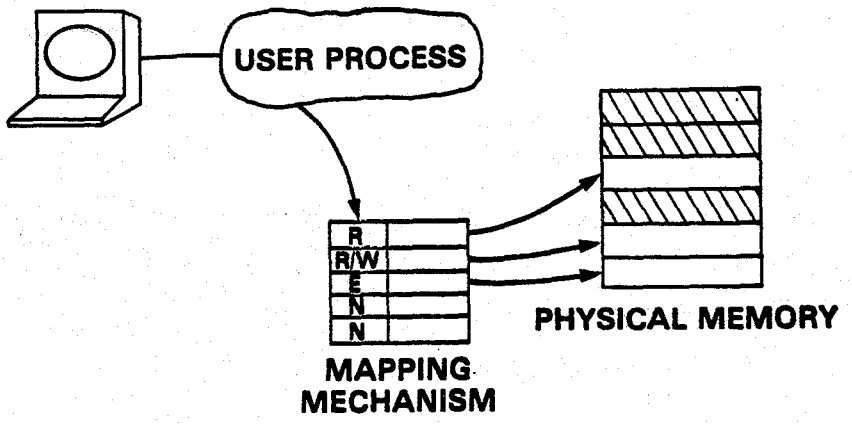
PROCESS SWITCHING



HARDWARE SUPPORT: PROCESS SWITCHING

ESSENTIAL:	ABILITY TO SUPPORT MULTIPLE PROCESSES
HIGHLY DESIRABLE:	HARDWARE INSTRUCTIONS TO AID PROCESS SWITCHING
CONVENIENT:	MULTIPLE SETS OF MECHANISM FOR MULTIPLE PROCESSES

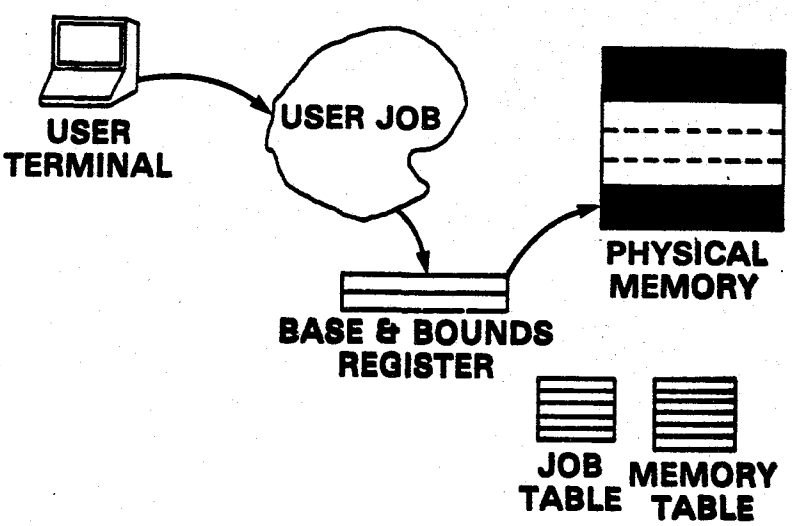
MEMORY ACCESS: SEGMENTED VIRTUAL MEMORY



EXPLICIT SEGMENTATION

- REALIZE MEMORY OBJECT ABSTRACTION
- PROVIDE INTERPRETIVE ACCESS
- VIRTUALIZE MEMORY & STORAGE
- DISTINCT READ/WRITE ACCESS
- SHARING
  - INTERPROCESS COMMUNICATION
  - BETWEEN USERS
- COMPATIBILITY CONSIDERATION
  - BASE & BOUND — SEGMENT
  - E.G., MULTICS EMBEDDED GCOS

MEMORY ACCESS: BASE REGISTER



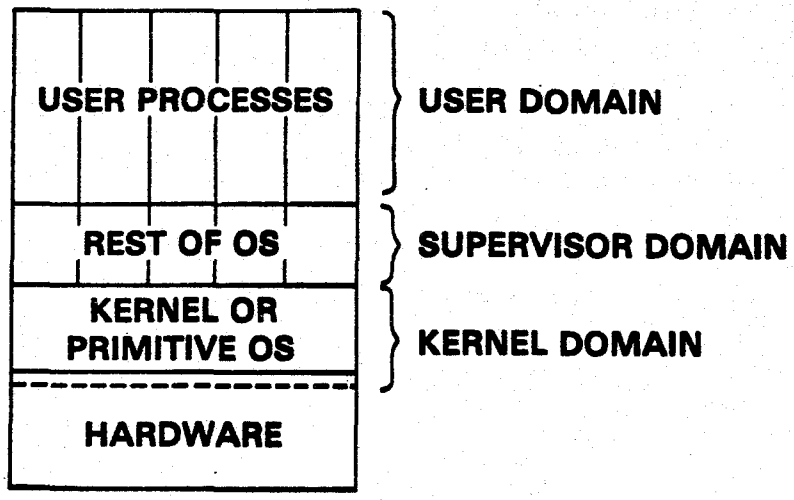
HARDWARE SUPPORT: SEGMENTATION

ESSENTIAL: PER-PROCESS SEGMENTED MEMORY
- HARDWARE MECHANISM
- R/W CONTROL BY ENTITY
- KERNEL CONTROL OF MECHANISM

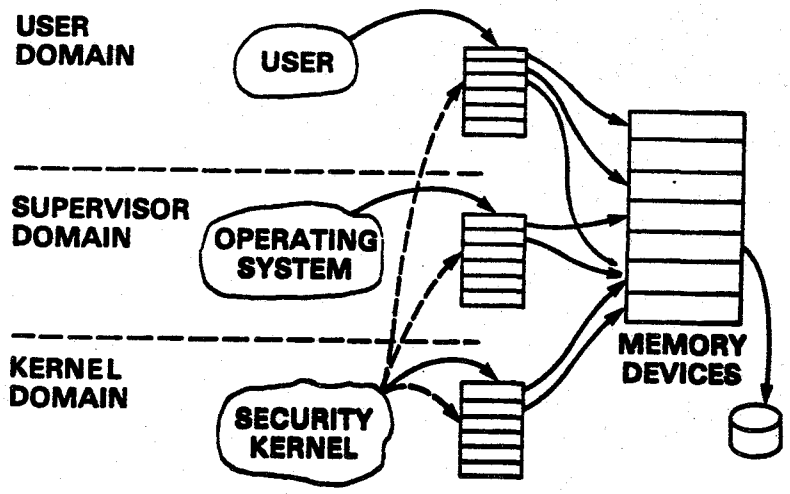
PROTECTION DOMAINS

- ENFORCE ISOLATION OF KERNEL
  - ACCESS TO KERNEL SEGMENTS
  - CROSS DOMAIN PARAMETERS
- FOR DISTRIBUTED KERNEL
  - DISTINCT DOMAINS IN EACH PROCESS
- APPLICABLE TO USER & OPERATING SYSTEM
  - PRIVILEGED MODE COMPATABILITY
  - PROTECT USER'S SERVICES
- HIERARCHICAL DOMAINS (RINGS) SUFFICIENT

OS AND KERNEL SUPPORT



PROCESS DOMAINS OF EXECUTION



HARDWARE SUPPORT: DOMAINS

ESSENTIAL:	TWO DOMAINS OF EXECUTION
	CONTROLLED ENTRY TO PRIVILEGED DOMAIN
HIGHLY DESIRABLE:	THREE HIERARCHICAL DOMAINS
	CONSISTENT ACCESS CONTROL

KERNEL ENGINEERING CRITERIA

- SECURITY KERNEL CONCEPT DEMONSTRATION
- SECURITY—MITRE ON PDP 11/45
- PERFORMANCE—MULTICS "AIM"
  - PROCESS SWITCHING
  - SEGMENTATION (WITH PAGING)
  - DOMAIN (RING) CROSSING
- FUNCTIONALITY/COMPATABILITY—AIR FORCE MULTICS
  - GENERAL PURPOSE UTILITY
  - INSTALLATION SECURITY PARAMETERS
  - USER INTERFACE OF MODEL
  - OPERATIONS
  - ADMINISTRATION

SUMMARY

- METHODICAL DESIGN PRINCIPLES
  - DEFINE "SECURE"
  - SPECIFY FUNCTIONS PRECISELY
  - USE MODERN HARDWARE & OPERATING SYSTEM
- ENGINEERING VIABILITY
- VERIFICATION SUITABILITY

This paper originally appeared in Technical Papers, USE Inc. Spring Conference, March 5-9 1979, p. 245-250.

SECURITY KERNELS: A METHODICAL DESIGN OF SYSTEM SECURITY

Roger R. Schell, Lt. Col., USAF
Assistant Professor of Computer Science, Naval Postgraduate School

INTRODUCTION

The security kernel is a relatively recent technical breakthrough for computer security. During the past seven years this technology has transformed the designer's game of wits with penetrators into a methodical design process with a predictably secure outcome. Controlled experiments with existing large and small computers have confirmed not only that a kernel can provide security, but also that it is practical in terms of performance, functional capability, and compatibility.

A secure computer system will not occur as the spontaneous result of other design goals. Security must be explicitly designed in from first principles. Our purpose here is to better understand techniques that are commonly used in the methodical design of a kernel-based system. In the interest of brevity we will not belabor the substantial mathematical and security validation aspects of the technology (available in the open literature [1]). Rather we will concentrate on the design implications.

BACKGROUND

Computer systems processing sensitive information fall into two categories with respect to security:

- In the first case, the computer and all its users are within a single security perimeter established by guards, dogs, fences, etc. Only these external security controls are required to maintain security. This case is not our immediate concern since no failure or subversion of the computer itself can compromise security.
- In the second case the computer itself must internally distinguish multiple levels of information sensitivity and user authorization. The internal security controls of hardware and computer programs must insure that each user may access only authorized information. The inability of contemporary computers to effectively provide such protection against repeated and undetected penetration is widely reported [2]. This is the problem we will address.

Relying on only external controls is in many cases undesirable because of the added expense and increased security risk from error-prone manual procedures. In addition, external controls cannot provide the secure sharing of information needed for many applications, such as integrated data bases and networks -- forcing us to forego many of the capabilities of modern computers. Fortunately, since my introduction [3] of the security kernel concept in 1972, this technology has matured into the means for demonstrably effective and practical internal controls.

THE SECURITY KERNEL DESIGN PROCESS

The underlying concept is that a small portion of the hardware and software (called a security kernel) can provide internal security controls that are effective against all possible internal attacks -- including those never

thought of by the kernel designers. This means that bugs or malicious attacks contained in applications, or even the operating system, absolutely cannot cause unauthorized access to information. But such a concept is of little practical value unless the designer can methodically proceed to successful implementation. We will now examine the major steps in the design process.

REFERENCE MONITOR ABSTRACTION

The security kernel is based on an underlying "security theory" for conceptualizing the idea of protection — applicable to people accessing documents as well as to a processor accessing memory. This has been formulated [4] as a reference monitor that facilitates active entities (e.g., people or their programs) called subjects making reference to passive entities (e.g., documents or computer files) called objects, based on a set of current access authorizations. In addition, the reference monitor facilitates subjects changing the access authorizations, again based on the current authorizations. Although important to the fundamental underpinnings of the security kernel, we will not further pursue the theoretical aspects.

However, what we can anticipate is that if the kernel is to provide protection, then it must of necessity actually implement the functions [5] of the reference monitor, and there is one important implication. Our formulation is fundamentally interpretive: that is, every reference to information (e.g., by a processor to core memory) must go through the security kernel. This observation highlights the need for attention to the question of performance, although, as we will see, the hardware component of a security kernel answers this very nicely.

SECURITY POLICY

It is external laws, rules, regulations, etc., that establish what access is to be permitted. In particular, a given system can only be said to be "secure" with respect to some specific policy. There are two distinct aspects of security policy.

- Non-discretionary (mandatory) policy externally constrains what access is permissible. In terms of the reference monitor, the idea is that we can label objects (information) to reflect sensitivity, and we can correspondingly label subjects (people) to reflect their authorizations. One of many examples is the Department of Defense (DoD) classification and clearance labels (secret, confidential, etc.). For such a policy the reference monitor (in our case the security kernel) must insure that access to classified information is always confined to cleared users. Most contemporary computer systems do not provide the labeling required to support non-discretionary policy — implicitly making all access permissible.
- Discretionary policy provides a finer granularity within (but cannot substitute for) the non-discretionary constraints; individual subjects can decide which of the permissible accesses will actually be allowed. Again, DoD provides an example with their "need to know" policy. Many computer systems permit users to specify what other users can access their files — to support a discretionary policy.

A significant design concern is whether we must have a distinct system design for the almost endless number of policies. Although several special systems have been built (especially for DoD), the current state of the art allows a

single, uniform mechanism for nearly all practical policies. A general purpose computer example is Honeywell's Multics [6] product with its access isolation mechanism (AIM) for non-discretionary policies and access control lists (ACLs) for discretionary policies. In the Multics example, General Motors, MIT, and the Air Force all use the same system but with different installation parameters to customize it to their quite different security policies.

The implication is that the kernel designer does not have to concern himself with the particular security policy of a specific customer. He must, however, consider the two broad classes of policy: discretionary and non-discretionary.

MATHEMATICAL MODEL

The notion of a mathematical model has been associated with the security kernel nearly since its inception [7]. A mathematical model is a powerful design tool for formally translating the requirements of security policy into a precise representation of the behavior of the corresponding security kernel. The model mathematically represents the state of a system, and prescribes the criteria for a secure state -- with respect to the policy classes being considered, of course.

The model also carefully defines a set of functions (or rules) for changing the state of the system and for permitting subjects to reference objects. These rules enforce certain properties. One is the rather interesting "confinement property" (also called the *-property in the literature, for historical reasons). This property insures that sensitive information can never be written into an object whose security label is incorrect, e.g., insures that DoD secret information cannot be written into an unclassified object.

The power of the model comes from the fact that a rather unusual set of rules have been discovered: it has been proven that if the initial state is secure, these rules can never produce a state which is insecure. This means that if the model indeed represents the behavior of the security kernel, then no use of the kernel can cause a compromise of information security. In other words, the model dictates what must and must not be included in the kernel. Furthermore, no other part of the system (e.g., operating system or application program) can violate security. The existing models reported in the literature (for example [8]) can be applied or used as the basis for a new model.

KERNEL SPECIFICATION

The mathematical model defines what functions the kernel must provide, but there are numerous choices of a specific design. In general we can think of the specification as defining a set of subroutines (supervisor calls) and hardware functions to implement the functions of the model. The kernel data bases will implement the model state. Formal specification techniques are attractive, and because of the small size of the kernel, they have been successfully used [9].

As with any design effort, preparing the specifications is a creative activity, molded by the peculiar design goals (other than security) of the

system. Yet there are several samples to serve as a guide: kernel specifications have been prepared for a DoD communications processor on an IBM minicomputer, a minicomputer time sharing system for the DEC PDP-11 and the Honeywell Level 6, a virtual machine monitor for the IBM 370, and Multics [10] for the Honeywell Level 68.

MECHANISMS FOR KERNEL IMPLEMENTATION

Past efforts to develop secure systems include several design projects where security was used as the major excuse for inclusion of exotic, incomplete or incompatible hardware and software mechanisms. Even when well-intentioned, the proposed mechanisms reflected the biases of the designers more than the logical result of the security structure. In contrast, the security kernel approach provides a careful basis for selecting appropriate mechanisms.

First we note that a successful implementation of a kernel is based on three [11] engineering principles: (1) completeness, in that all accesses to information must go through the kernel; (2) isolation, in that the kernel must be tamperproof; and (3) verifiability, in that there must be a direct correspondence to the model and specification requirements. These three principles and the underlying reference monitor abstractions of subject and object determine the desirable mechanisms. The issue of efficiency (performance) guides the choice of hardware versus software realization of the mechanisms.

The abstraction of a subject is realized in the kernel through a process. Since many security policies relate authorizations to people, a process will often serve as a surrogate for a user. Therefore, the kernel and its associated operating system will generally provide a computational structure of distinct, communicating processes. Performance considerations may well dictate hardware support for rapid process exchange for the central processors.

A subject will generally have access to several objects with distinct security attributes (e.g., security labels and read/write permissions). The fundamentally interpretive nature of the reference monitor (reflected in the completeness principle) requires that the distinct attributes be visible at the time of actual reference to information. This implies an explicitly segmented memory. In any general programming system, efficiency will absolutely dictate that this kernel function be implemented in segmentation hardware, at least for the CPU, and possibly for I/O as well.[1]

The isolation principle will generally require that the kernel operate in its own protection domain. The common desire to additionally separate the operating system and applications leads to a total of three hierarchical domains -- rather than the traditional two (e.g., user and supervisor mode). The restricted domain mechanism commonly called protection rings are sufficient, and hardware implementation [12] is straightforward and efficient when segmentation hardware is available. More complex general domain machines or capabilities machines will also work, but offer little advantage to the kernel design.

With respect to performance, the hardware support is very important to a kernel design because of its fundamentally interpretive nature. Several of

the past and ongoing kernels have experienced significant performance degradation because of the lack of sufficient hardware. On the other hand, all the indicated hardware capabilities are well understood (see for example [13]) and have proven themselves in working commercial products. Although most current machines do not include all these hardware capabilities, advances in modern microelectronics have made them economically available for most future computers. Furthermore, the needed hardware can be provided as smooth extensions of most existing architectures without introducing fundamental incompatibilities. Thus we see that no exotic or unproven hardware or software mechanisms are needed for implementation of a security kernel.

SUMMARY

The security kernel technology is the only currently available technology that can provide both the required internal security and functional capabilities. First, we have a firm technical foundation on which to construct a security kernel for a specific system. We know that the resulting kernel will support an unusually wide range of commercial and governmental information protection policies. Available mathematical models precisely define the functional requirements of the kernel and provide definitive criteria for establishing the security of the resulting system -- to essentially any degree of confidence and mathematical rigor required.

Secondly, we have the software and hardware techniques for a kernel-based system with good performance. Operating systems with the necessary clean, explicit structure of cooperating, asynchronous processes are also recognized for their flexibility and power. Segmentation hardware for efficient mediation of access to information is available for everything from single chip microcomputers to large scale, general purpose multiprocessors. Similarly, hardware protection rings to isolate (protect) the kernel and operating system are available and well understood. Both are attractive for reasons other than security, and microelectronics have made them very affordable.

Finally, we have the empirically validated engineering concepts needed to apply the security kernel without degradation of system capabilities -- for everything from a specialized communication switch to a computer utility. The kernel for security is compatible with a wide range of common architectures, so that an existing software base can be generally preserved. Not only can basic compatibility be preserved, but also the kernel lends itself to an orderly, progressive evolution of a well-structured existing system towards a kernelized version. In addition, there are guidelines for defaults and installation parameters so that a kernel-based system to support a specific security policies can be introduced into a facility without major disruption.

In short, designing a secure system does require a substantial and conscious effort, perhaps over an extended period. The security kernel approach provides for the orderly and methodical completion of an efficient and capable system with truly effective security for computerized information.

REFERENCES

1. Harper, S. R., "ESD Computer Security Annotated Bibliography," MCI-76-10, The MITRE Corporation, December 1976.

2. Schell, R. R., "Computer Security: The Achilles' Heel of the Electronic Air Force?", Air University Review, January-February 1979 Vol. XXX No. 2, p. 16-33.

3. Schell, R. R., "Abstract of a Virtual Memory Security Kernel," a participant's position paper provided in private correspondence, 5 May 1972, to Chairman of the IEEE Workshop on Privacy and Protection in Operating Systems (13 and 14 June 1972).

4. Rhode, R. D., "ESD 1976 Computer Security Development Summary," MCI-76-2, The MITRE Corportation, January 1977.

5. Schell, R. R., "Effectiveness -- The Reason for a Security Kernel," Proc. National Computer Conference, 1974, p. 975 and 976.

6. Whitmore, J. C., et al, "Design for Multics Security Enhancements," ESD-TR-74-176, December 1973.

7. Schell, Roger. R., "Notes on an Approach for Design of Secure Military ADP Systems," In Preliminary Notes on the Design of Secure Military Computer Systems, USAF Electronics Systems Division MCI-73-1, January 1973, p. 1-1 through 1-5 (prepared remarks for Panel Session, "Privacy and Protection in Operating Systems," Proc. ACM Annual Conference, August 1972, p. 665 and 666.)

8. Bell, D. E., and LaPadula, L. J., "Computer Security Model: Unified Exposition and Multics Interpretation," ESD-TR-75-306, June 1975.

9. Schiller, W. L., "The Design and Specification of a Security Kernel for the PDP-11/45," ESD-TR-75-69, May 1975.

10. Schroeder, M. D., et al, "The Multics Kernel Design Project," Proc. Sixth ACM Symposium on Operating Systems Principles, November 1977, p. 43-56.

11. Anderson, J. P., "Computer Security Technology Planning Study," ESD-TR-73-51, October 1972.

12. Schroeder, M. D., and Saltzer, J. H., "A Hardware Architecture for Implementing Protection Rings," Comm. ACM 15,3 (March 1972), p. 157-170.

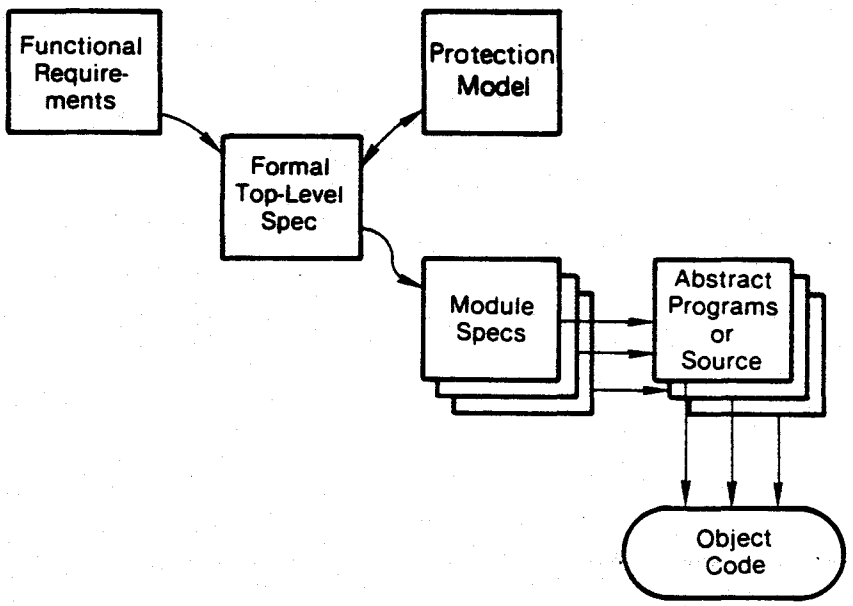
13. "Secure Communications Processor Architecture Study," ESD-TR-76-351, 1976.

Peter Tasker
MITRE Corporation

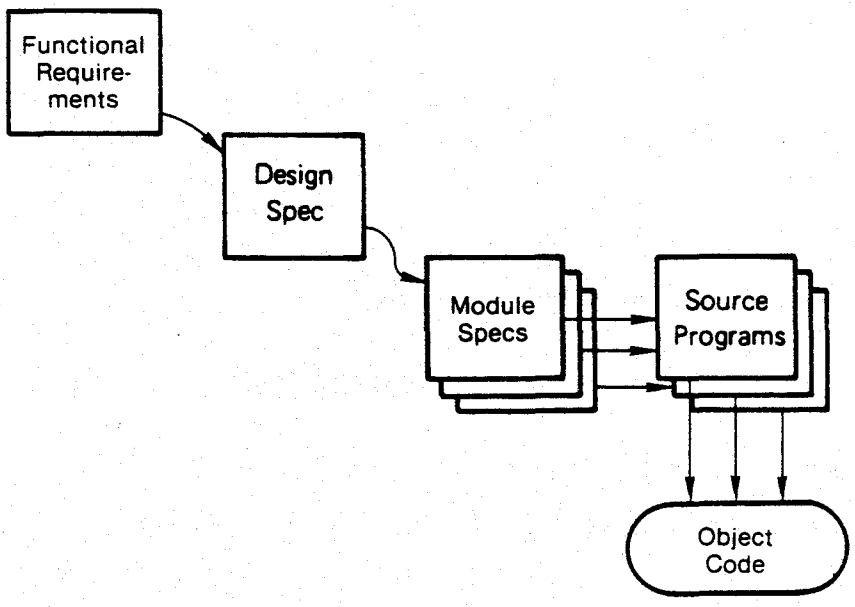
Formal Specification And Verification

- Formal Specifications
- Top-Level Design Verification
- Detailed Design And Verification
- Program Verification

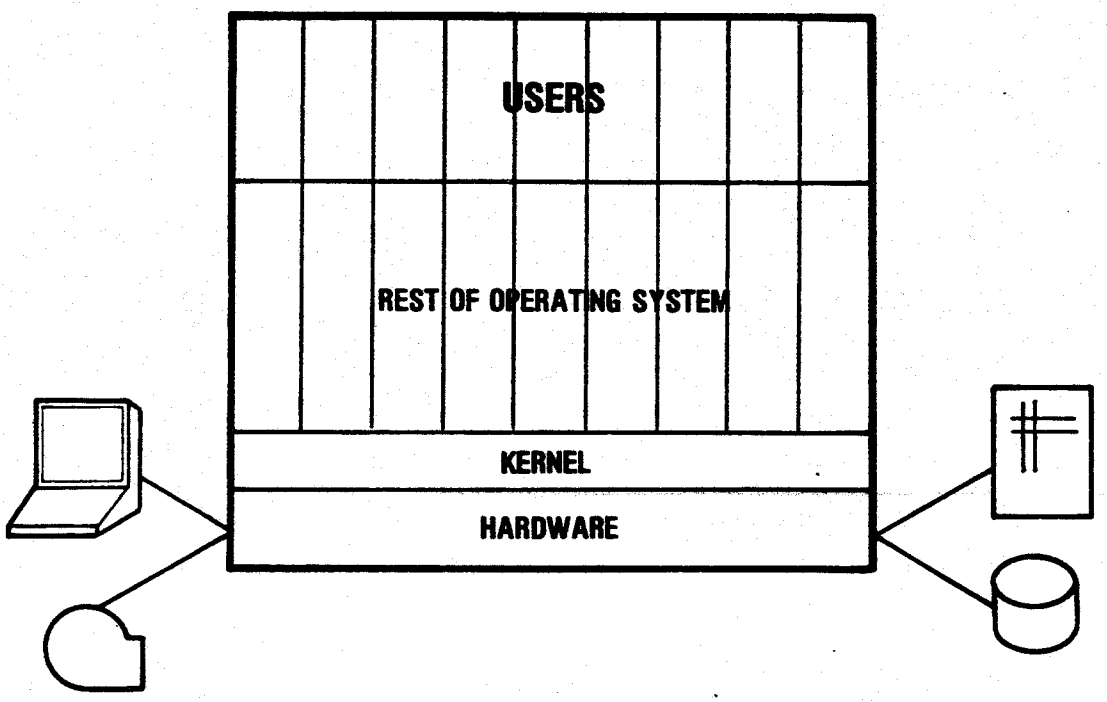
FORMAL SPECIFICATION IN SOFTWARE DEVELOPMENT



TRADITIONAL SOFTWARE SYSTEM DEVELOPMENT



PROTECTION IN A KERNEL-BASED SYSTEM



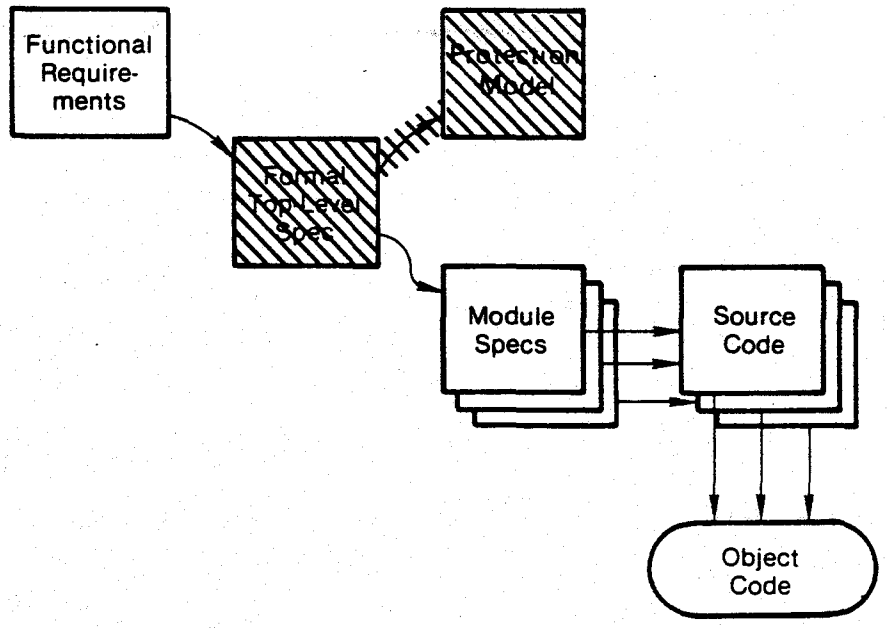
GENERIC TOP LEVEL KERNEL OPERATIONS

PROCESS	FILE	DEVICE
CREATE	CREATE	(IF NOT TREATED AS A FILE)
DESTROY	READ	READ
SWAP/CHANGE	WRITE	WRITE
SEND MSG	GET/OPEN	RESERVE
RECEIVE MSG	RELEASE/CLOSE	RELEASE
	GIVE	
	RESCIND	

FORMAL TOP-LEVEL SPECIFICATION

- FINITE STATE MACHINE REPRESENTATION OF A SPECIFIC SYSTEM
- FORMAL LANGUAGE
- TOP-LEVEL: ONLY OPERATIONS AND EFFECTS "VISIBLE TO THE USER"

TOP-LEVEL VERIFICATION OF TRUSTED SYSTEMS



MODEL AXIOMS TO BE PROVEN TRUE

SIMPLE SECURITY CONDITION

READ: SECURITY-LEVEL (SUBJECT) $\geqslant$ SECURITY-LEVEL (OBJECT)

*–PROPERTY

WRITE: SECURITY-LEVEL (OBJECT) $\geqslant$ SECURITY-LEVEL (SUBJECT)

ACTIVITY, TRANQUILITY AND ERASURE PRINCIPLES

MESSAGE MODULE SPECIFICATION

OFUN send(receiver)

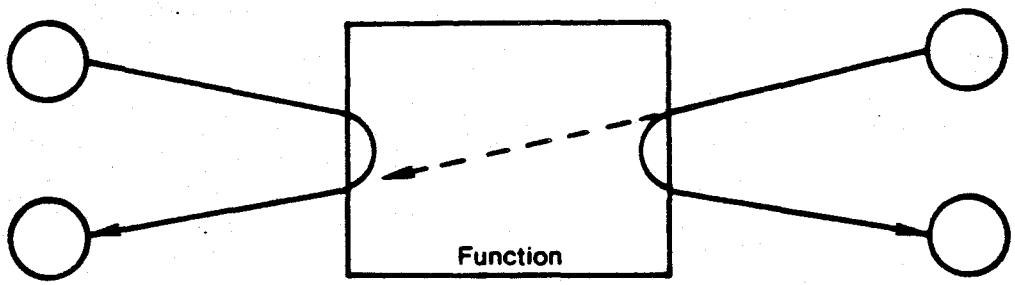
EXCEPTION

KEnoAccess: security_class(caller) > security_class(receiver)
KEmailboxBusy: message_seg(receiver) $\neq$ null

EFFECT

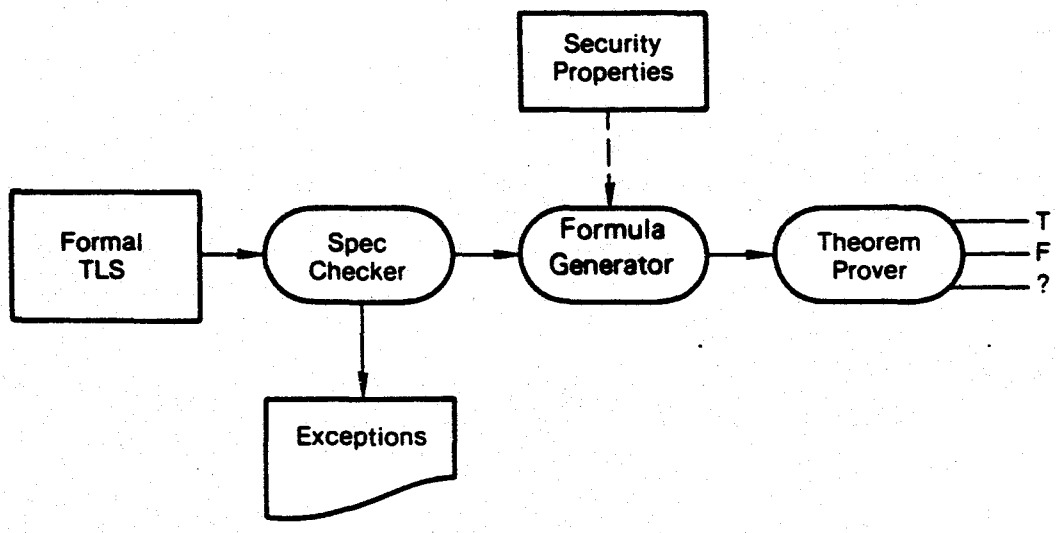
'message_seg(receiver) = message_seg(caller)
'message_seg(caller) = null

SECURITY VERIFICATION: TLS



Prove: no direct or indirect insecure flows on object-pair basis

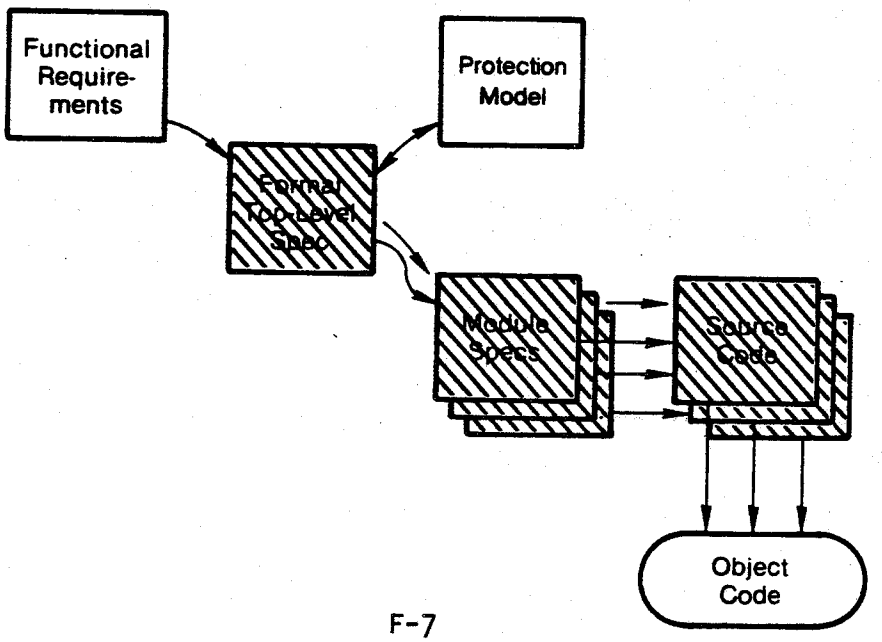
AUTOMATED SECURITY VERIFICATION



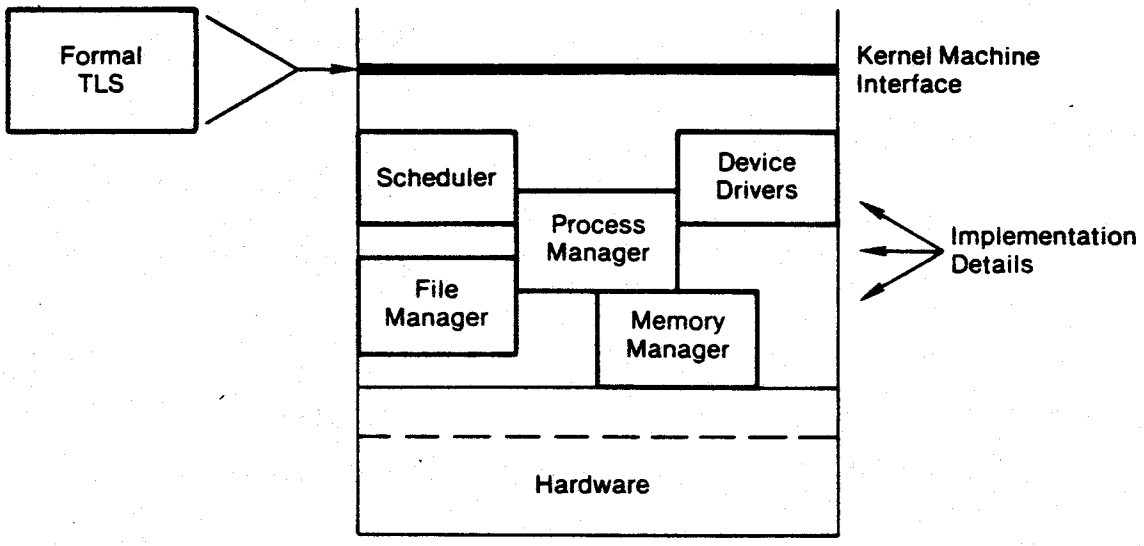
Verification Tools

	SPECIFICATION	THEOREM PROVER
SDC	INAJO	SCHORRE
SRI	SPECIAL	BOYER-MOORE
ISI	AFFIRM	AFFIRM
TEXAS	GYPSY	BLEDSOE

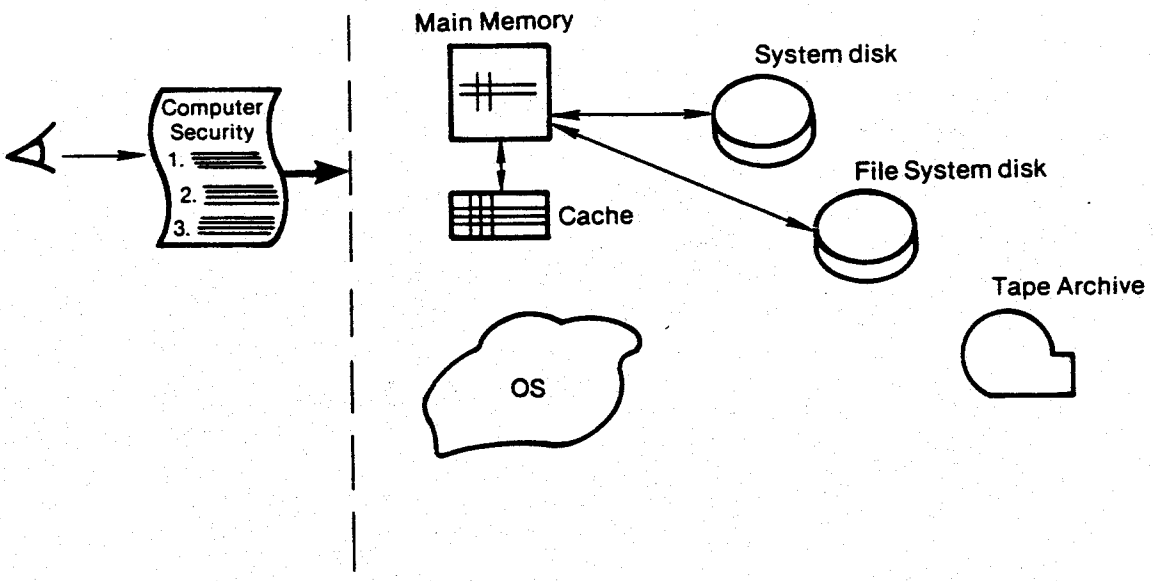
DETAILED DESIGN OF TRUSTED SYSTEMS



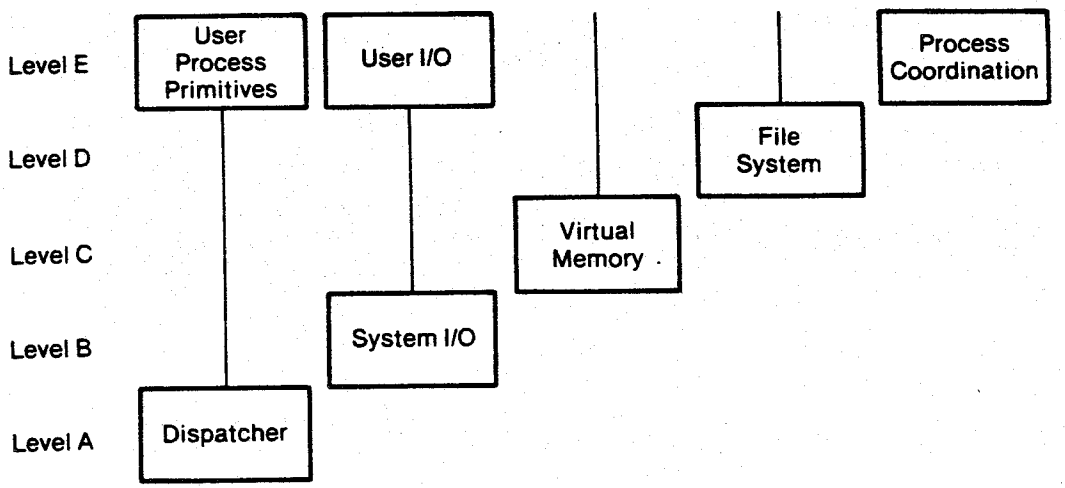
TOP LEVEL SPECIFICATION



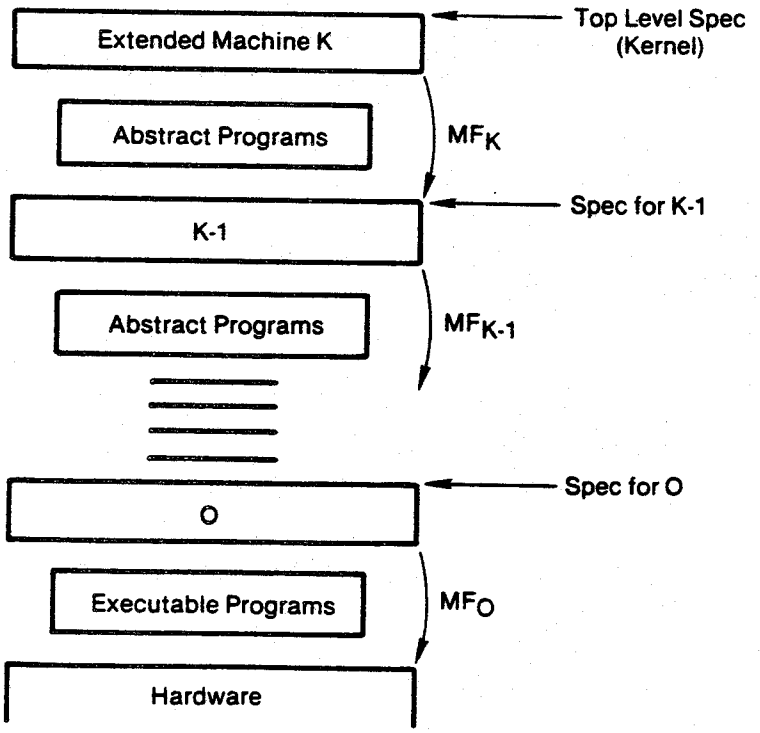
A SEGMENT ABSTRACTION



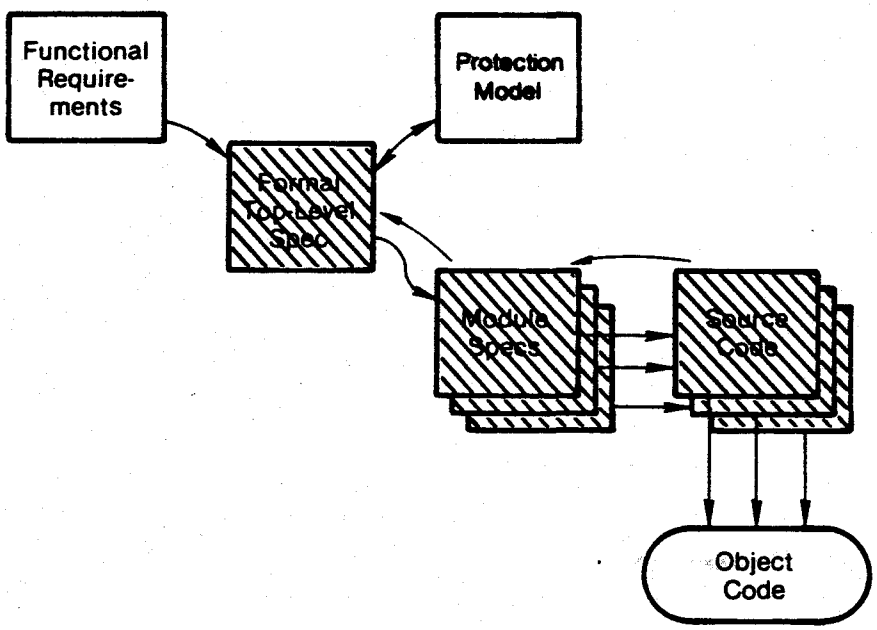
SYSTEM DECOMPOSITION (TACEXEC)



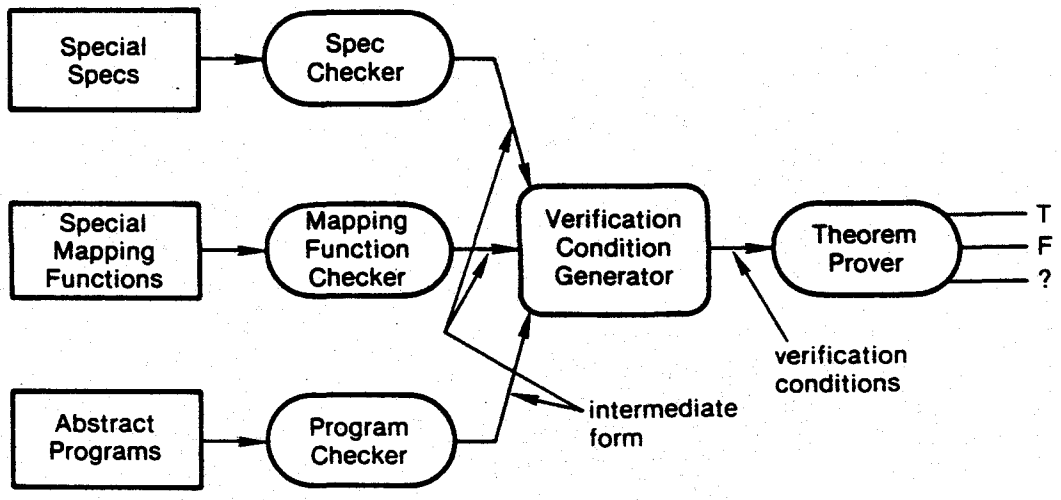
RELATING FORMAL TLS TO IMPLEMENTATION



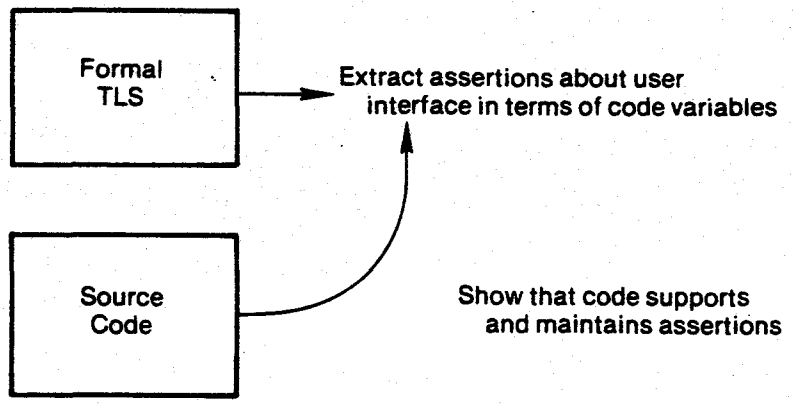
DESIGN AND VERIFICATION OF TRUSTED SYSTEMS



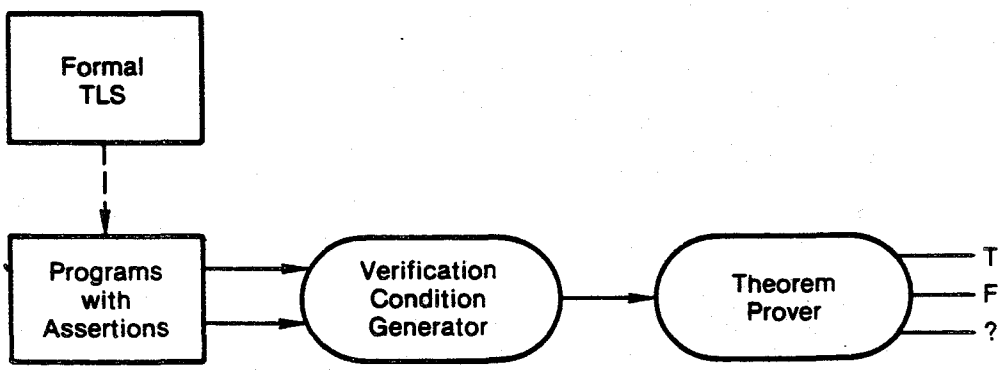
SRI IMPLEMENTATION PROOF TOOLS



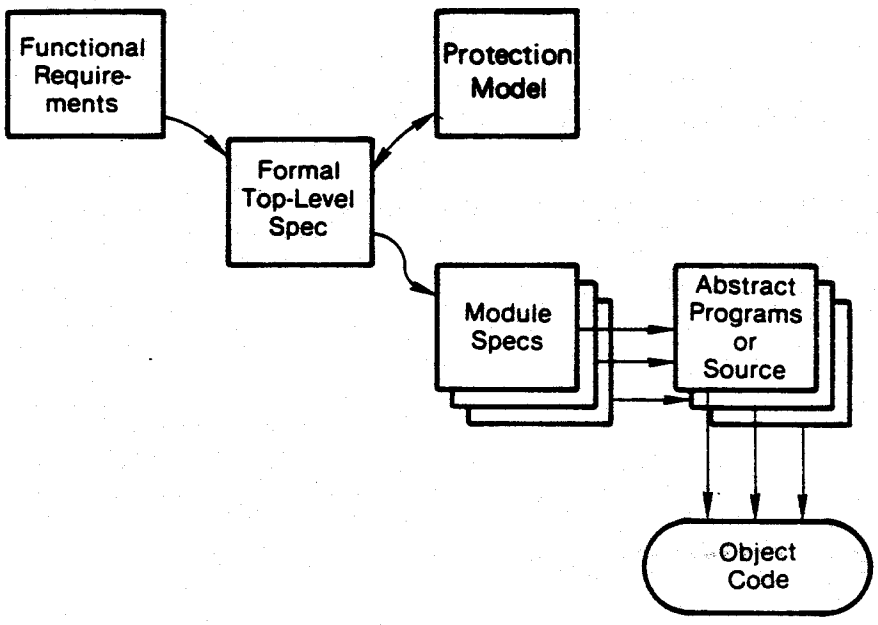
MANUAL VERIFICATION



AUTOMATED PROGRAM VERIFICATION



FORMAL SPECIFICATION AND VERIFICATION



DOD
Kernelized Secure Operating System
(KSOS)

Dr. E. J. McCauley
Ford Aerospace & Communications Corporation
Palo Alto, CA

KSOS Topics

- Project Goals
- Design Methodology
- KSOS Design
  - Kernel
  - UNIX Emulator
  - Non-Kernel System Software
- Application Considerations

PROJECT GOALS

Ford Aerospace & Communications Corporation

KSOS REQUIREMENTS SUMMARY

- Provable security: based on security Kernel and trusted processes
- UNIX compatibility
- Efficiency comparable with UNIX
- Administrative support features
- General purpose Kernel
  - Multiple machines
  - Emulators for other operating systems
  - Non-UNIX applications

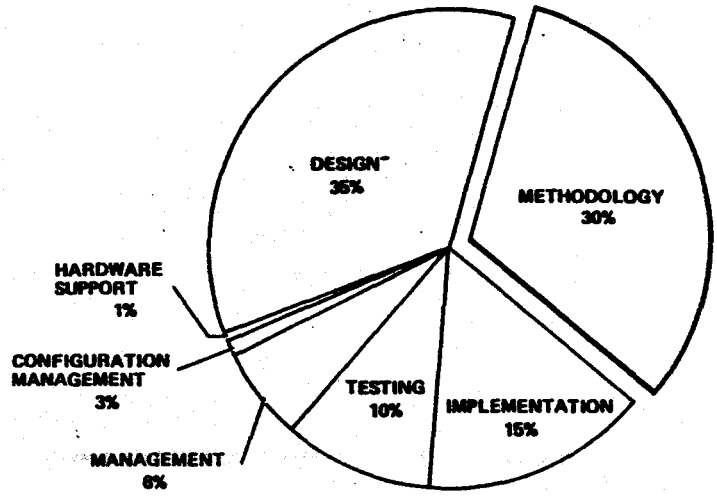
Ford Aerospace & Communications Corporation

Influences on the KSOS Design

- External Requirements
- Hardware Limitations (PDP-11/70)
  - Process switching costly
  - Memory management hardware
  - Absolute addressing in device registers
- Design Methodology

DESIGN METHODOLOGY

KSOS RESOURCE ALLOCATION



Ford Aerospace & Communications Corporation

METHODOLOGY

Things which make KSOS different

Costs: 30%

Benefits:

- Confidence in security of system
- Reduced support costs (?)
- Landmark practical application

HDM VS. CLASSICAL DESIGN METHODS

DESIGN STAGE	HDM	"CLASSICAL"
REQUIREMENTS DEFINITION	FORMAL MATHEMATICAL MODEL OF SECURITY	BROAD STATEMENT OF SECURITY REQUIREMENTS
FUNCTIONAL ALLOCATION	HIERARCHICAL DECOMPOSITION OF SYSTEM INTO LAYERS OF VIRTUAL MACHINES	DECOMPOSITION INTO FUNCTIONS PERFORMED BY EACH CPCI
FUNCTIONAL SPECIFICATION	FORMAL MATHEMATICAL SPECIFICATION IN A NON-PROCEDURAL LANGUAGE (SPECIAL)	B5 SPECIFICATIONS: INTERFACES, INPUT, PROCESSING AND OUTPUT FOR EACH FUNCTION

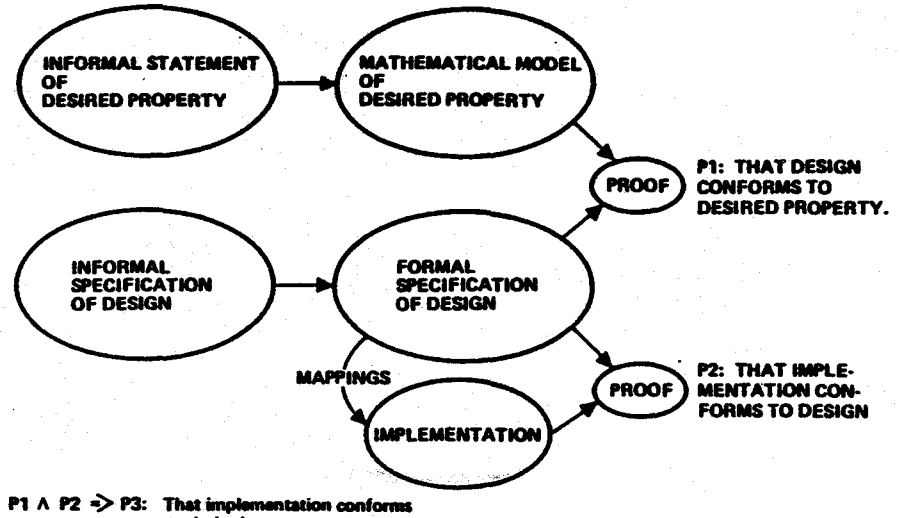
Ford Aerospace & Communications Corporation

HDM VS. CLASSICAL DESIGN METHODS (Continued)

DESIGN STAGE	HDM	"CLASSICAL"
DETAILED DESIGN	DATA REPRESENTATIONS, ABSTRACT PROGRAMS	STRUCTURED ENGLISH, FLOW CHARTS, ETC.
IMPLEMENTATION	VERIFIABLE LANGUAGE	LANGUAGE CHOSEN FOR EFFICIENCY, MAINTAINABILITY, COMPATIBILITY, ETC
REQUIREMENTS COMPLIANCE	FORMAL PROOFS: DESIGN VS SECURITY MODEL	MANUAL REVIEWS: FUNCTIONAL CONFIG. AUDIT
	CODE VS DESIGN	PHYSICAL CONFIG. AUDIT

Ford Aerospace & Communications Corporation

SCHEMA FOR CONSTRUCTION OF PROVABLE SYSTEMS



P1 ∧ P2 => P3: That implementation conforms to desired property.

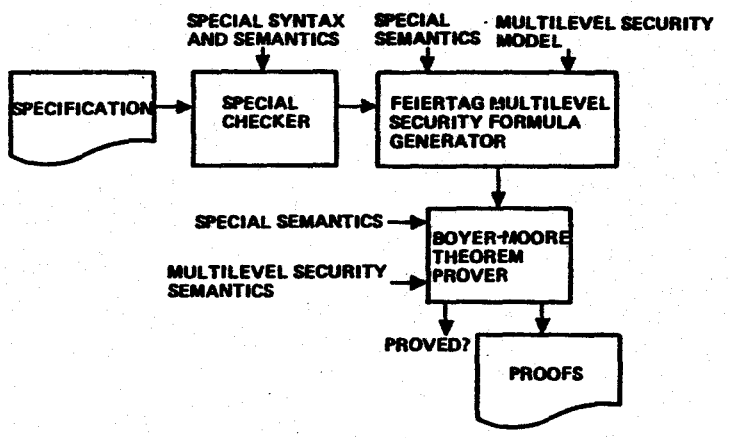
Ford Aerospace & Communications Corporation

KSOS PROOF GOALS

- Prove KSOS conforms to multilevel security model
- Total automation of proof process
- Complete design proof
- "Illustrative" implementation proofs

Ford Aerospace & Communications Corporation

DESIGN PROOF ENVIRONMENT



Ford Aerospace & Communications Corporation

DESIGN PROOF RESULTS

- KSOS Kernel design proven to be multilevel secure.
- Statistics
  - 505 formulas generated
  - 322 proved by formula generator
  - 183 proved by theorem prover
  - 15 KL10 CPU-minutes required

Ford Aerospace & Communications Corporation

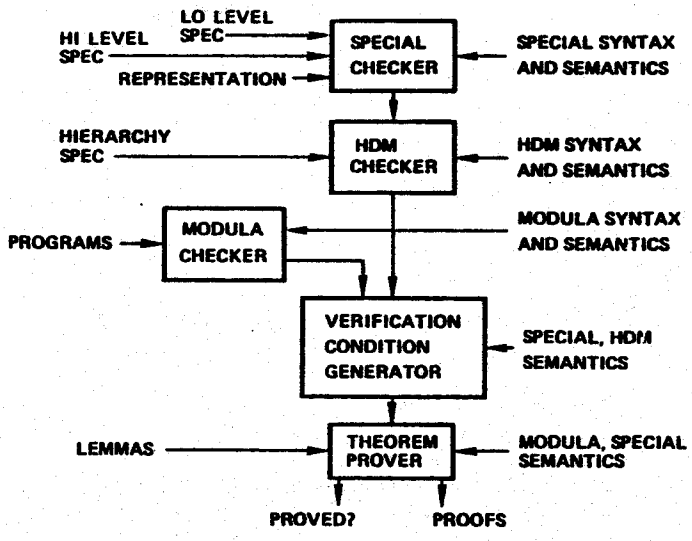
KSOS IMPLEMENTATION LANGUGE

Choice →

- Euclid
- Modula
- Gypsy
- Pascal
- C
- Ada

Ford Aerospace & Communications Corporation

IMPLEMENTATION PROOF ENVIRONMENT

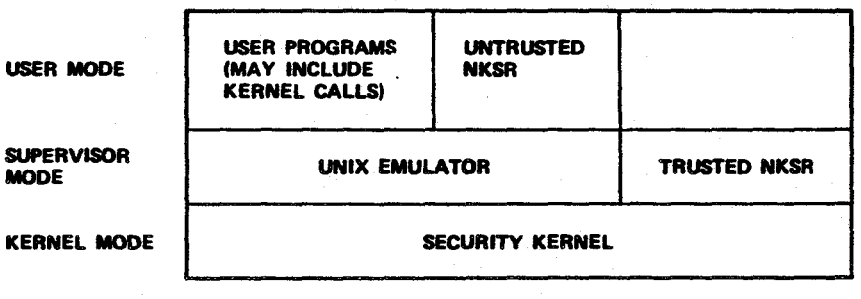


Ford Aerospace & Communications Corporation

KSOS DESIGN

Ford Aerospace & Communications Corporation

FUNCTIONAL COMPONENTS



(NKSR: NON-KERNEL SECURITY RELATED SOFTWARE)

Ford Aerospace & Communications Corporation

KSOS Kernel Objects

Processes	Program in execution
Process Segments	Portions of virtual memory of a process
Files	Linear array of data blocks, "flat" name space
Devices	Special type of file
Subtypes	Encapsulation tool

Ford Aerospace & Communications Corporation

Kernel Objects

Every object has:

A Name (Secure Entity IDentifier, "SEID",)

Type Independent Information

- Owner (user and group)
- Security classification (e.g. TOP SECRET)
- Security compartment set (e.g. NOFORN, caveats)
- Integrity classification
- Integrity compartment set (now always null)
- Discretionary access information

Ford Aerospace & Communications Corporation

KSOS Kernel Process

- Cheap, plentiful
- May be privileged K_invoke, K_spawn
- K_fork: "cloning"
- Inter-Process Communication
  - messages
  - shared segments

Ford Aerospace & Communications Corporation

KSOS Kernel Process Segments

- Variable sized
- Rendezvous with shared segments by names
- Options for system designer
  - normal
  - stickey
  - locked

Ford Aerospace & Communications Corporation

KSOS Kernel Files and Devices

- "Flat" name space
- Linear array of data blocks
- Single file up to 300 Mbytes (600?, 1200?)
- Mountable volumes, fully protected

THE PROBLEM

How can the Kernel aid in insuring the integrity of higher level constructions like UNIX Directories without knowledge of their internal structure and semantics.

SUBTYPES

- Kernel knows some files are "SPECIAL"
- Each subtype has discretionary access for all files of that subtype.
- Triple Open Condition
  - Mandatory security and integrity
  - Discretionary access to subtype
  - Discretionary access to file

Ford Aerospace & Communications Corporation

KSOS UNIX Emulator

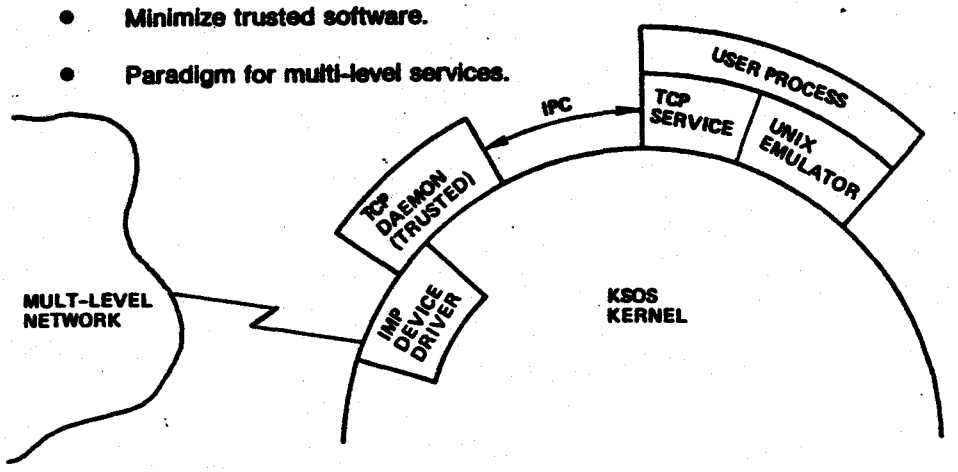
UNIX System call → Transformation → Requests for Equivalent Kernel Services

- Defined by two interfaces
- Directory manager: K_spawn'ed for directory writes
- Most of TCP support.

Ford Aerospace & Communications Corporation

KSOS Network Interface

- Minimize trusted software.
- Paradigm for multi-level services.



Ford Aerospace & Communications Corporation

KSOS Non-Kernel System Software

- Secure User Services
  - Login, logout, change working level, change level of a file.
- System Operation Services
  - TCP Daemon, secure mail, line printer spooler, mount/unmount.
- System Maintenance Services
  - File system maintenance, system generation.
- System Administrative Services
  - Control of users, privileged software installation, auditing.

Ford Aerospace & Communications Corporation

SUMMARY

- Combination of proven software engineering and formal methods
- Provably secure
- Faithful emulation of UNIX
- Usable Kernel
- Full administrative support for turnkey use

Ford Aerospace & Communications Corporation

KERNELIZED VM-370 OPERATING SYSTEM (KVM)

Marvin Schaefer
System Development Corporation

KVM/370: A SECURITY RETROFIT OF VM/370

B. D. Gold, R. R. Linde, R. J. Peeler

M. Schaefer, J. F. Scheid, P. D. Ward

System Development Corporation
Santa Monica, California 90406

KVM/370 GOALS

KVM/370 is the retrofit of VM/370 with a verifiable security kernel.

KVM/370 should:

- Enforce DoD security policy
- Preserve compatibility with existing VM/370 applications
- Salvage a maximum of existing VM/370 CP code
- Have reasonable efficiency compared against periods processing applications
- Protect against penetration attacks

TYPES OF SECURITY VIOLATIONS

- Machine Takeover (obtain real supervisor state)
- Data Theft (unauthorized access to data)
- Direct Write-Down
- Indirect Write-Down
- (not addressed) Denial of Service

KVM/370 SECURITY POLICY

- The kernel restricts the access by subjects to objects.
- Subjects are programs and processes.
- Objects are pages and I/O devices, both virtual and real.
- Directories and spool files are protected across discontinuities.
- Protection is provided between distinct security levels.
- Enforcement of two properties
  - The Basic Security Principle
  - The *- Property

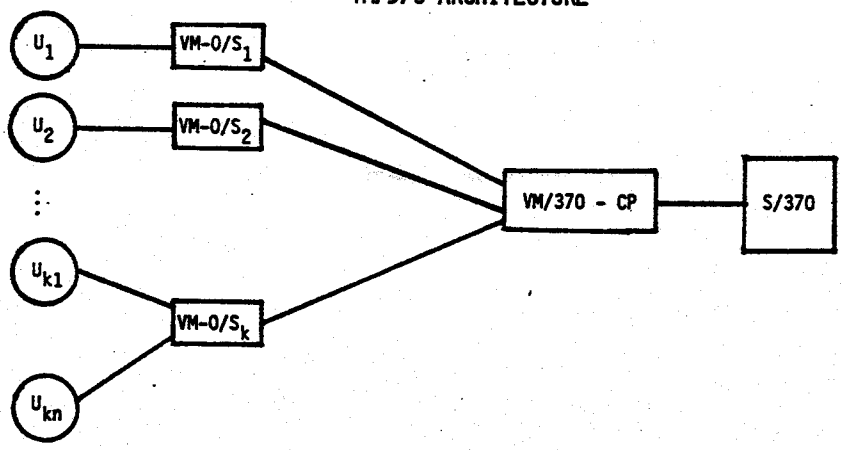
CLASSES OF PENETRATIONS ELIMINATED IN KVM/370

- DATA SECURITY VIOLATIONS
  - Asynchronous Parameter Replacement
  - Bizarre I/O Requests
- CONFINEMENT VIOLATIONS
  - Direct Write-Down
  - Data Buried in "Innocent" Communications
  - Covertly Shared Variables

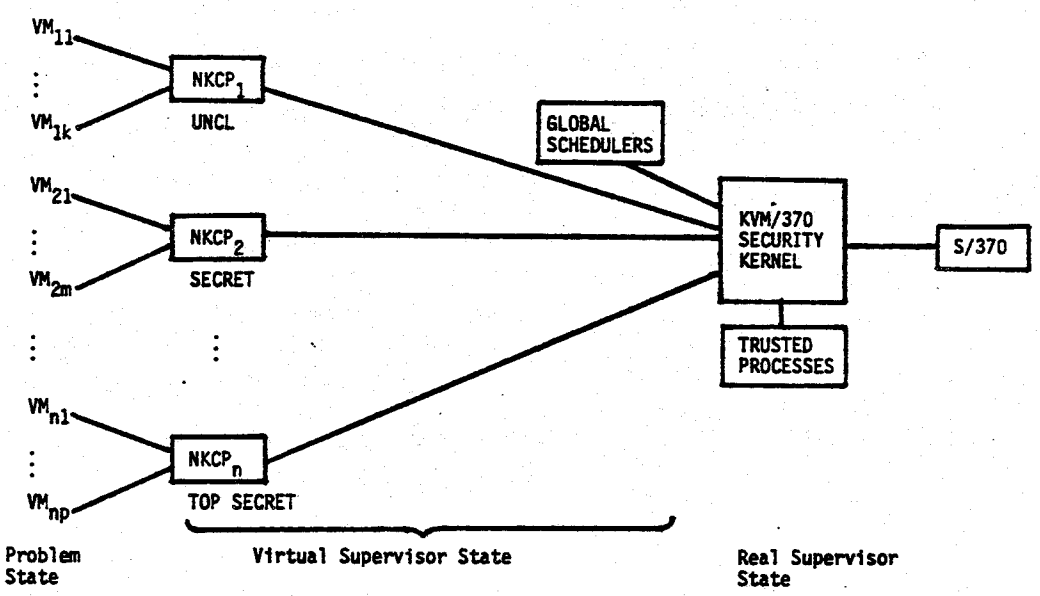
COVERT CONFINEMENT VIOLATION EXAMPLES

- "INNOCENT" Communications
  - Accounting
  - Error Recording
  - Semaphores
- COVERT SHARED VARIABLES
  - Time to Complete a Request
  - Resource Exhaustion
  - Order of Completion of Tasks
  - Page Selection

VM/370 ARCHITECTURE



KVM/370 ARCHITECTURE



KVM/370 DOES MORE THAN CORRECT VM/370 SECURITY FLAWS

- THE KVM/370 ARCHITECTURE IS DIFFERENT FROM THAT OF VM/370
- PATCHING KNOWN ERRORS DOES NOT GUARANTEE CERTIFIABLE SECURITY
- A "HARDENED" VM/370 WOULD NOT PROVIDE MULTI-LEVEL DATA SHARING
- CONVERT AND CLANDESTINE COMMUNICATION CHANNELS WOULD PERSIST IN "HARDENED" VM/370

KVM/370 SECURITY RETROFIT PRINCIPLES

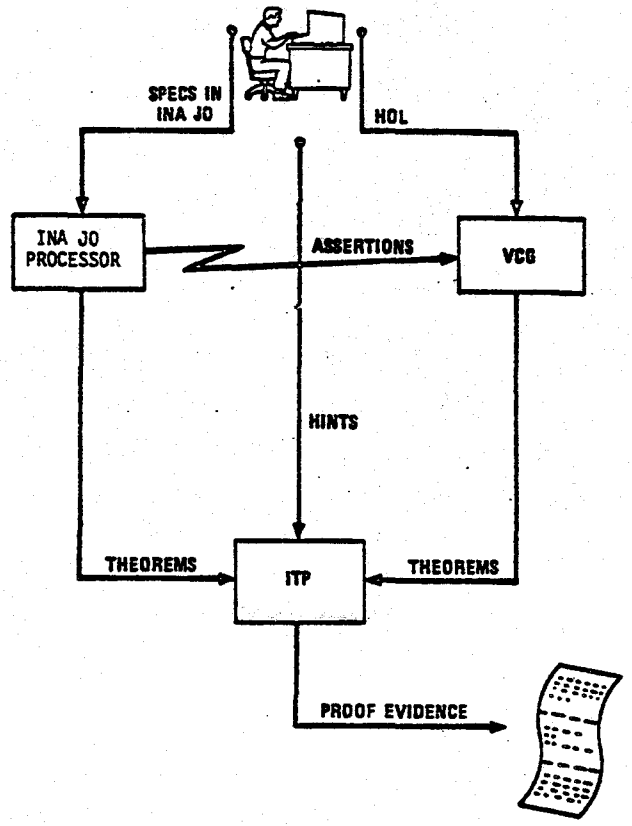
- DESIGN MUST ADHERE TO PRINCIPLES OF
  - LEAST PRIVILEGE
  - LEAST COMMON MECHANISM
- SECURITY RELEVANT CODE MUST BE TRUSTED
- UNIFORM PROTECTION MECHANISMS MUST BE USED

System Development Corporation

IDENTIFICATION OF SECURITY-RELEVANT MODULES

- FUNCTIONALLY-DEFINED GLOBAL MODULES LEAD TO SECURITY PROBLEMS
- DATA-TYPE ABSTRACTION MODULES RESOLVE SECURITY PROBLEMS
  - * INFORMATION HIDING
  - * CONSISTENCY CONTROL
  - * COMMON MECHANISM ISOLATION

APPROACH



KVM/370 FORMAL DESIGN PROCESS

- MODULAR DECOMPOSITION
  - * NON-PROCEDURAL "ENGLISH" DESCRIPTIONS
- FORMAL VERIFIABLE SPECIFICATIONS
  - * SECURITY MODEL IN CORRECTNESS CRITERION
  - * TOP-LEVEL ABSTRACTION
  - * INA JO SPECIFICATION METHODOLOGY AND TOOLS
    - - STRONG DATA TYPING
    - - MATHEMATICAL RIGOUR
    - - TOOL-ENFORCED REFINEMENT
    - - INTERACTIVE THEOREM PROVER
- HIGH ORDER LANGUAGE IMPLEMENTATION
  - * SUBSET JOVIAL J3
  - * CODING TEMPLATES
  - * CORRESPONDENCE TO SPECIFICATIONS

KVM/370 SECURITY KERNEL

- Interrupt Driven
- Controls
  - All Real I/O
  - All Paging & Spooling I/O
  - Allocation of
    - * DASD Pages
    - * Storage Pages
    - * DASD Spooling Cylinders
    - * I/O Devices

TRUSTED PROCESSES

- Long-Term Scheduling
- Authorization Process
- Directory Maintenance
- Unit Record Device Allocation
- Operator Process
- Accounting

GLOBAL SCHEDULERS

- Short & Medium Term Scheduling
- Allocates
  - Non Preemptive CPU time among security levels
  - Spooling Cylinders
- Schedules
  - Real I/O Devices
  - Real I/O Controllers
  - Real I/O Channels
- Selects Pages for Replacement
- Provides Centralized Error Recording

KVM/370 SCHEDULE

- March 1976 -- Feasibility Study Phase
- March 1977 -- Detailed Formal Design Phase
- May 1978 -- Implementation Phase
- December 1978 -- Integration Testing Phase
- September 1979 -- Prototype Testing, Tuning and Evaluation
- October 1980 -- Release of KVM/370

FY/80 THRUSTS

- KVM MATURATION
  - * TEST-SITE IMPLEMENTATIONS
    - DCEC
    - AFWL
    - IBM-FSD
  - * TIMING AND TUNING
  - * FEATURE ENHANCEMENTS
- SITE-SPECIFIC INSTALLATIONS
  - * MASS-STORE DEVICE SUPPORT FOR AFWL ICCS
- FORMAL SPECIFICATION VERIFICATION
  - * TOP LEVEL SPECIFICATION
  - * REFINED SPECIFICATION

SECURE COMMUNICATIONS PROCESSOR (SCOMP)

MATTI KERT

Honeywell

SCOMP HISTORY

- STARTED AS SECURE FRONT END PROCESSOR DEVELOPMENT FOR MULTICS UNDER PROJECT GUARDIAN
- CURRENTLY TWO SEPARATE DEVELOPMENT EFFORTS
  1. ARPA SPONSORED NAVELEX CONTRACT
  2. HONEYWELL INTERNAL DEVELOPMENT

OBJECTIVES

- PRIMARY OBJECTIVE

  DEVELOPMENT OF ADD-ON HARDWARE TO SIMPLIFY SECURITY KERNEL AND ITS VERIFICATION

- COMPLEMENTARY OBJECTIVES

  1. KERNEL SOFTWARE
  2. SOFTWARE DEVELOPMENT TOOLS
  3. UNIX EMULATOR

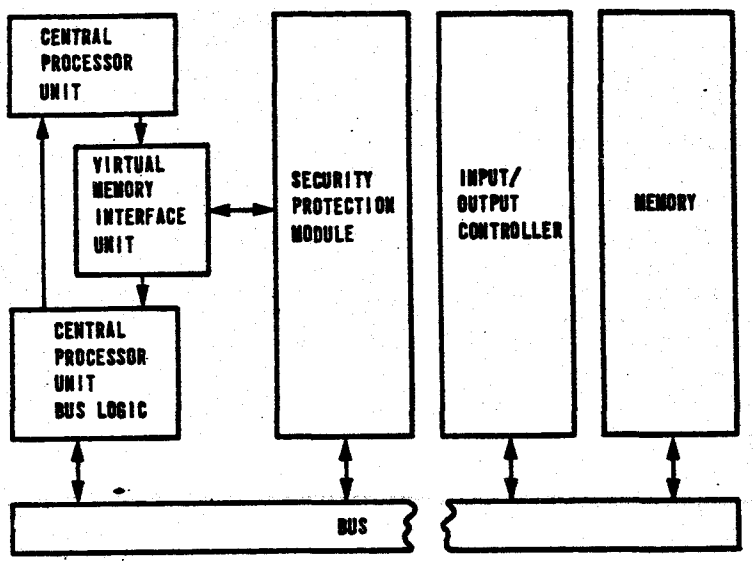
SCOMP HARDWARE OVERVIEW

- SCOMP HARDWARE CONSISTS OF A STANDARD MINICOMPUTER (HONEYWELL LEVEL 6) ENHANCED BY A SECURITY PROTECTION MODULE (SPM)

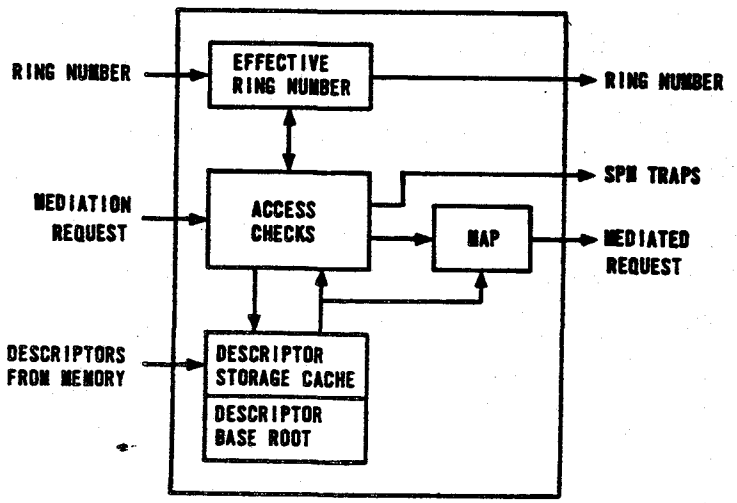
- FEATURES

  - MULTICS-LIKE RING STRUCTURE
  - RING CROSSING SUPPORT INSTRUCTIONS
  - MEMORY MANAGEMENT AND I/O MEDIATION
  - MILLION WORD ADDRESS SPACE
  - PAGE FAULT RECOVERY SUPPORT
  - FAST PROCESS SWITCHING

SPM + LEVEL 6 MINICOMPUTER = SCOMP



SPM BLOCK DIAGRAM



MEMORY ORGANIZATION

- PROCESS
  - ➡ DESCRIPTOR BASE ROOT
    - ➡ DESCRIPTOR TREE
- THREE LEVEL DESCRIPTORS SUPPORTING
  - SEGMENTS
  - PAGES
  - PAGED DESCRIPTOR SEGMENTS
- MEMORY ACCESS CONTROL AT ANY LEVEL
  - READ, WRITE, EXECUTE
  - RING BRACKETS
- SEGMENTS ≤ 2048 WORDS
  PAGES ≤ 128 WORDS

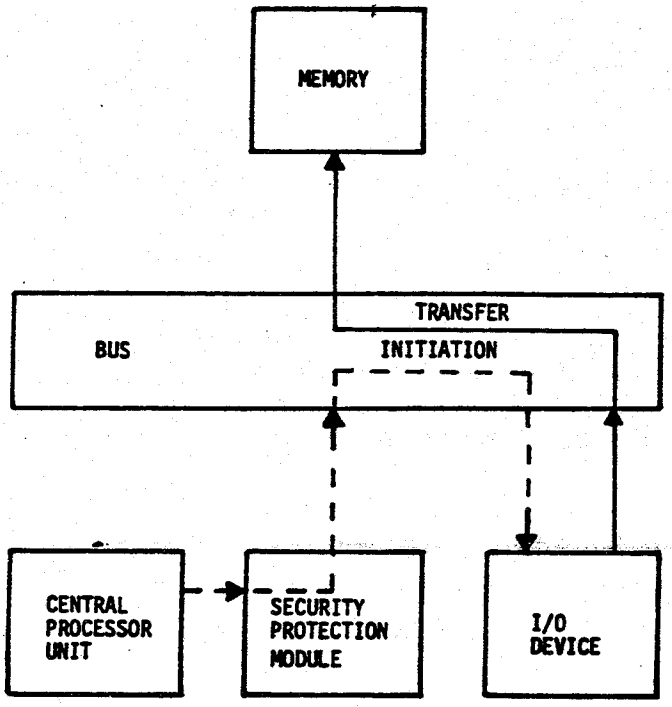
I/O MEDIATION

- MEDIATES CPU TO DEVICE AND DEVICE TO MEMORY
- I/O DEVICE DESCRIPTORS AND MEMORY DESCRIPTORS
- "PROGRAMMED I/O" FOR CONTROL AND STATUS

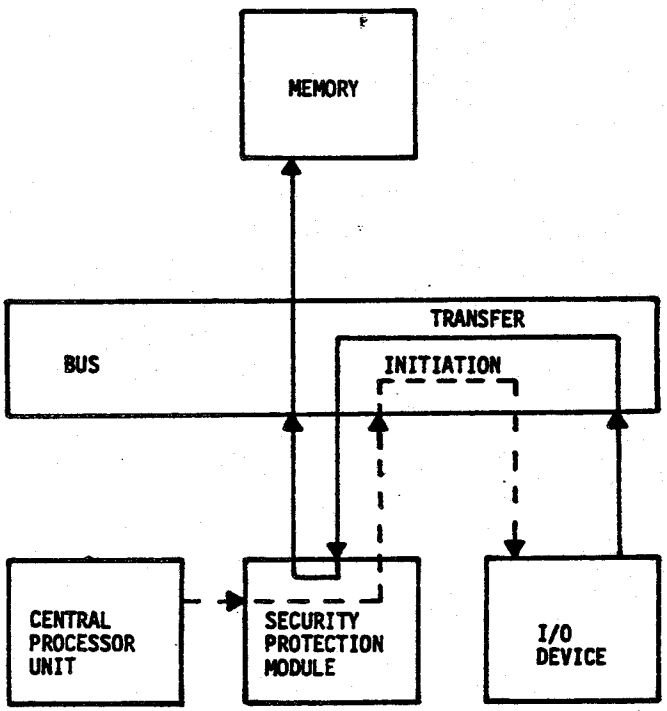
  TRANSFER AND FUNCTION CODES BOTH MEDIATED
- "DIRECT MEMORY ACCESS" (DMA) FOR DATA TRANSFER

  TRANSFER (READ/WRITE) BY I/O CONTROLLER MEDIATED

PREMAPPED I/O FLOW



MAPPED I/O FLOW



HARDWARE DEVELOPMENT STATUS

- SPM DEVELOPMENT MODEL OPERATIONAL IN FIRST HALF OF 1978
- SPM PRODUCTION PROTOTYPE IN TEST, FUNCTIONAL TEST COMPLETED IN JUNE 1979

SOFTWARE OVERVIEW

KERNEL

- FORMALLY SPECIFIED IN SRI's SPECIAL
- RELATIVELY SIMPLE KERNEL

  OBJECTS: SEGMENTS, DEVICES, PROCESSES
  NONDISCRETIONARY ACCESS CONTROL: SECURITY, INTEGRITY
  DISCRETIONARY ACCESS CONTROL: UNIX, RING BRACKETS, SUBTYPES
- DETAILED DESIGN IN PROCESS
- WILL BE CODED IN UCLA PASCAL

UNIX EMULATOR

- PRELIMINARY DESIGN COMPLETE
- WILL BE CODED IN C-LANGUAGE

SUMMARY

- HARDWARE SUPPORTED I/O MEDIATION
- HARDWARE/FIRMWARE SUPPORT TO MINIMIZE PROCESS SWITCHING OVERHEAD
- GENERAL MEMORY MANAGEMENT CAPABILITY THAT INCLUDES SUPPORT FOR DEMAND PAGING
- RELATIVELY SIMPLE KERNEL - TERMINAL I/O AND FILE SYSTEM OUTSIDE KERNEL

SECURE SYSTEMS APPLICATIONS

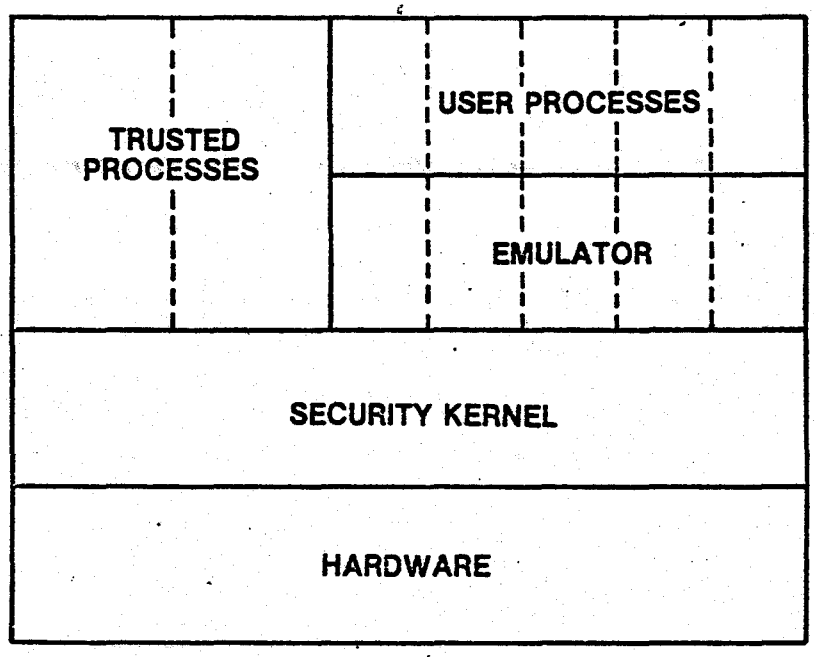
John P. L. Woodward
MITRE Corporation

APPLICATIONS FOR MULTILEVEL SECURE OPERATING SYSTEMS

BACKGROUND

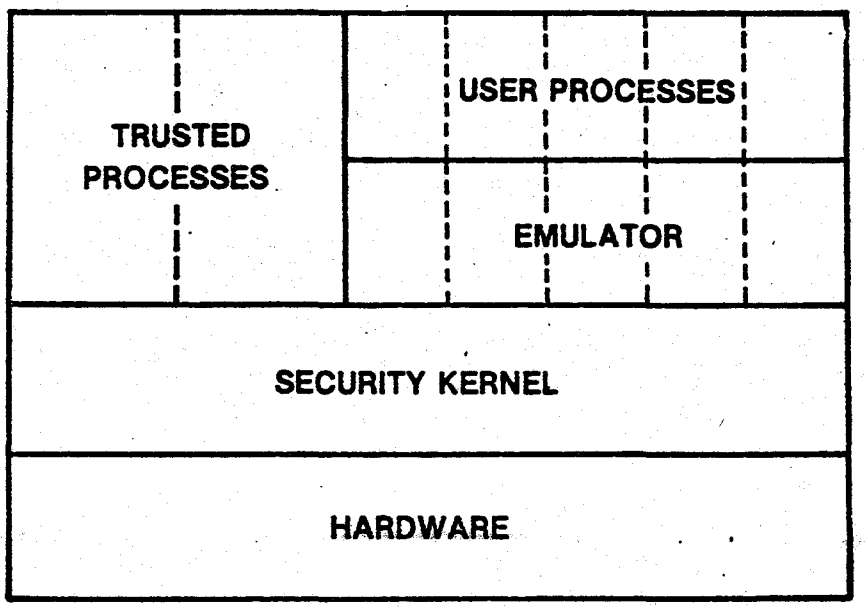
- NEED: RELIABLY CONTROL ACCESS TO MULTIPLE CLASSIFICATIONS OF DATA
  - COMPLETE ISOLATION (UNILEVEL)
  - CONTROLLED SHARING (MULTILEVEL)
- SOLUTION: THE SECURITY KERNEL APPROACH

SECURE OPERATING SYSTEMS



- DATA SEPARATION/ACCESS CONTROL
- ABILITY TO RUN VERIFIABLE (TRUSTED) APPLICATION CODE

UNILEVEL APPLICATIONS

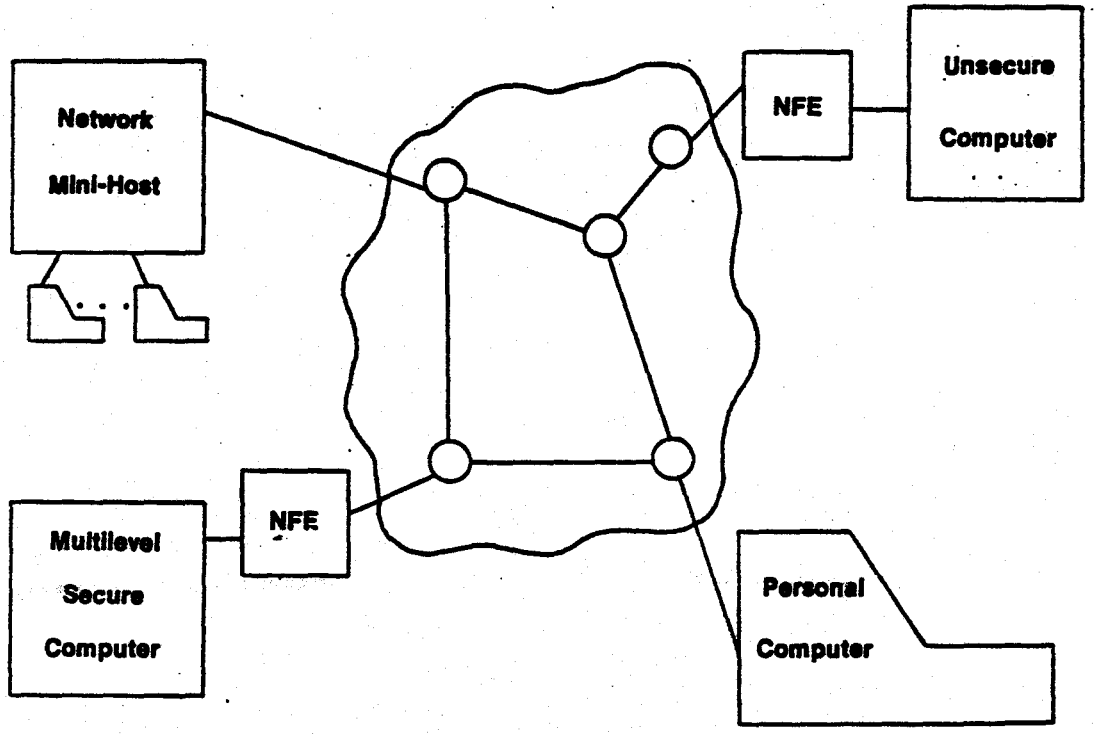


- ELIMINATES: MULTIPLE COMPUTERS
  PERIODS PROCESSING
  SYSTEM HIGH OPERATION
- NO ROBUST ACCESS TO DATA AT DIFFERENT CLASSIFICATIONS

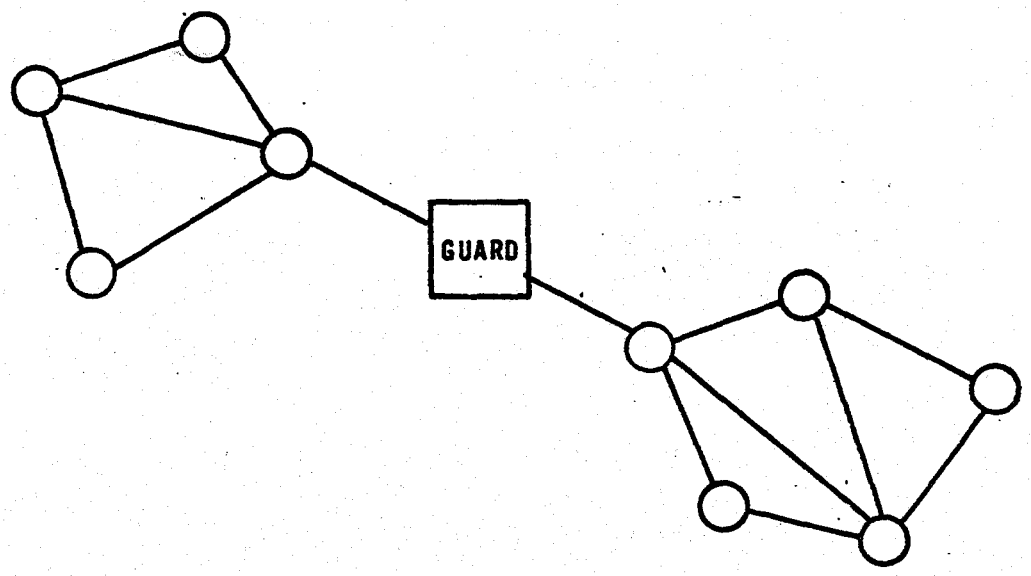
MULTILEVEL APPLICATIONS

- Address Need For A User To Process Multiple Levels
- Take Advantage Of Ability To Run Trusted Application Code
- Examples:
  - -Network Applications
  - -ACCAT Guard
  - -Database Management Systems
  - -Message Processing Systems

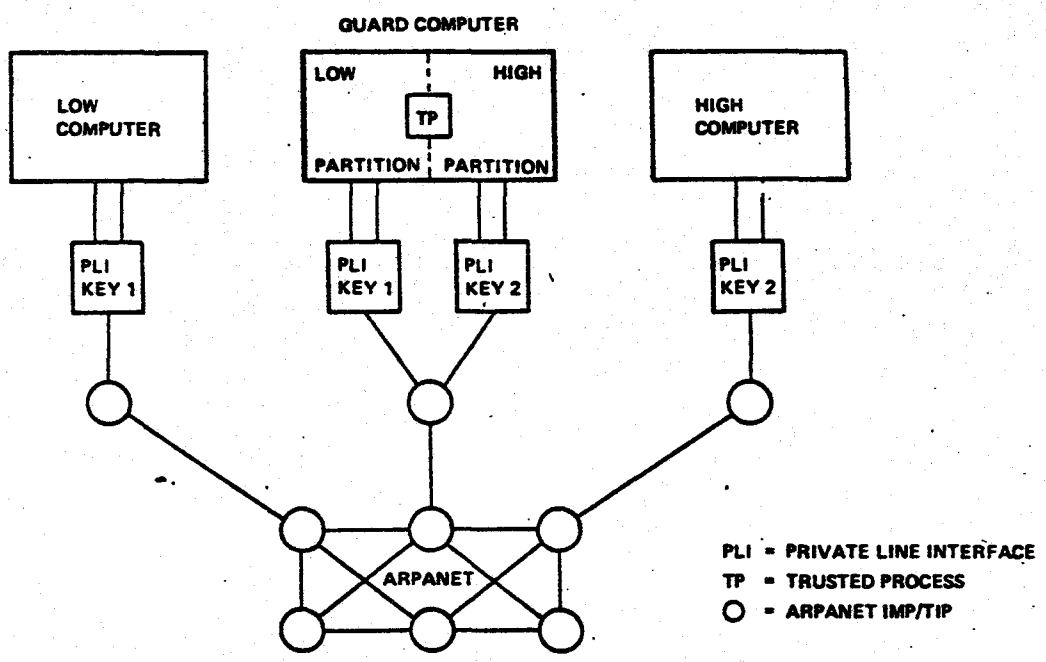
NETWORK APPLICATIONS



ACCAT GUARD SYSTEM



ACCAT GUARD PHYSICAL CONNECTIONS

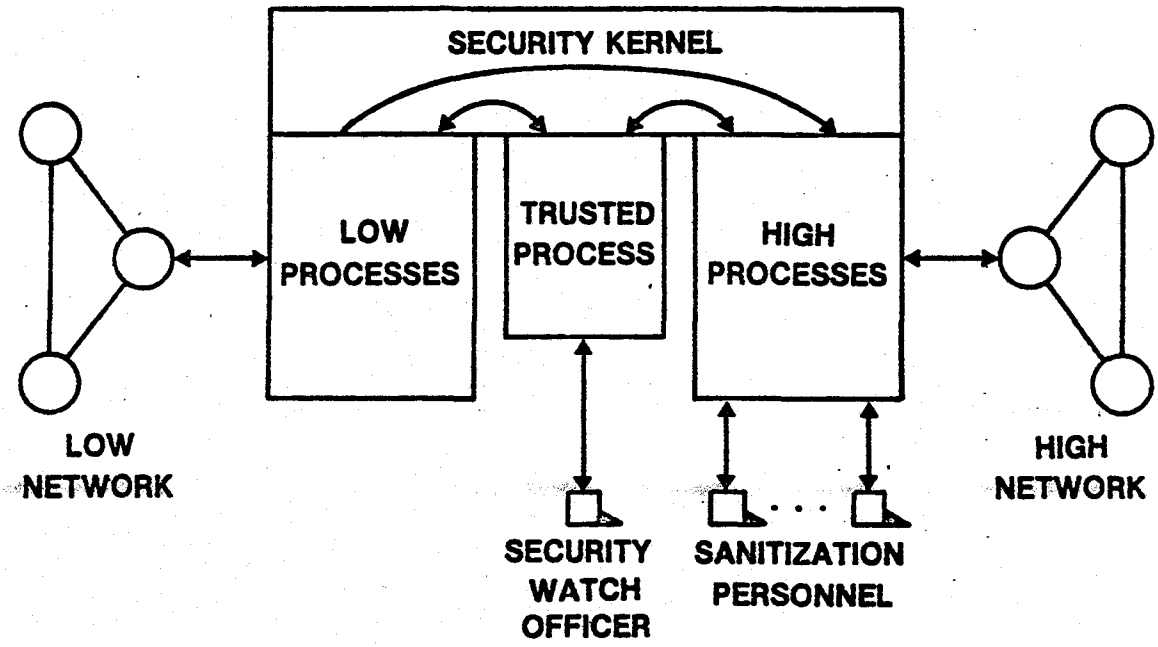


ACCAT GUARD IMPLEMENTATION

- RUNS ON KSOS
- LOW AND HIGH PROCESSES USE UNIX *tm EMULATOR
- TRUSTED PROCESS RUNS DIRECTLY ON KERNEL

ACCAT GUARD FUNCTIONALITY

- DATA TRANSFERS
  - DATABASE QUERIES/RESPONSES
  - MAIL



INTER-NETWORK MAIL

- Low To High
  - -No Security Issue
  - -No Human Intervention
- High To Low
  - -Possibility Of Compromise
  - -Review By Security Watch Officer
  - -Rejected Mail Feedback

DATABASE QUERIES

- Four Types: { Low To High / High To Low } { English / Canonical }

- High To Low Query
  - -English: Translated To Canonical Form
  - -Canonical: Security Watch Officer Review

- Response (Low To High)
  - -No Security Issue
  - -No Human Intervention

DATABASE QUERIES

- Low To High Query
  - -English: Translated To Canonical
  - -Canonical: No Human II Intervention

- Response (High To Low)
  - -Sanitized By Sanitization Personnel
  - -Reviewed By Security Watch Officer

SECURE DATABASE MANAGEMENT SYSTEMS

- ALLOW MEANINGFUL ORGANIZATION OF DATA AND PRESERVATION OF CLASSIFICATION
- RELY TO VARYING DEGREES ON PROTECTION OF UNDERLYING OS
- MAJOR ISSUES
  - BEST PLACE TO BUILD PROTECTION
  - NATURE OF HUMAN INTERFACE
- PAPER SURVEYS SEVERAL DESIGN EFFORTS

PREVIOUS SECURE DMS WORK

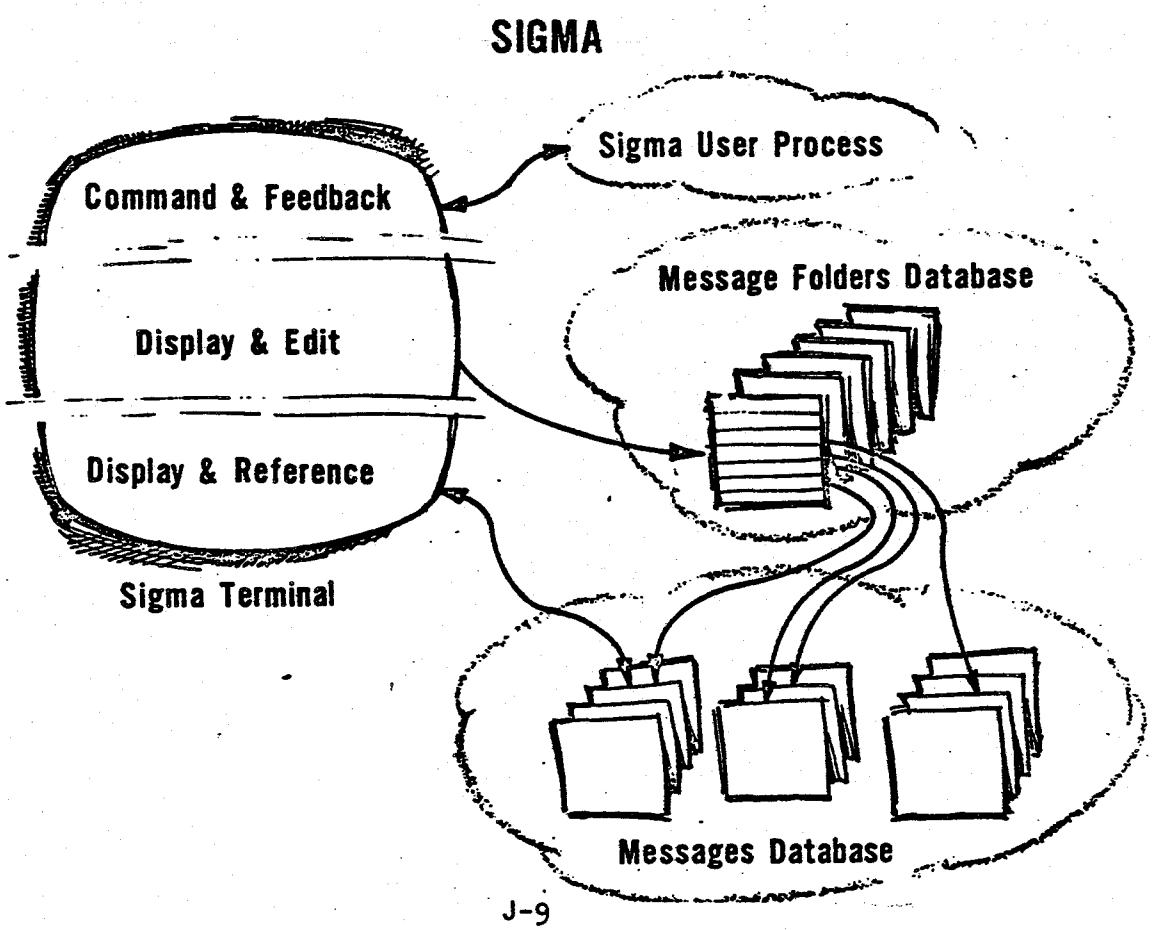
- SDC Secure Data Management System
  - -Based On Secure Multics OS
  - -Modelling And Design Effort
  - -DMS Has No Security-Relevant Code
- I.P. Sharp Protected DMS Tool
  - -Investigated The Design Of Kernel Primitives
  - -Family of Secure DMS
- MITRE Secure Ingres
  - -Similar To SDC Approach
  - -No Security-Relevant Code
  - -Cumbersome User Interface

MME SYSTEM FUNCTIONS

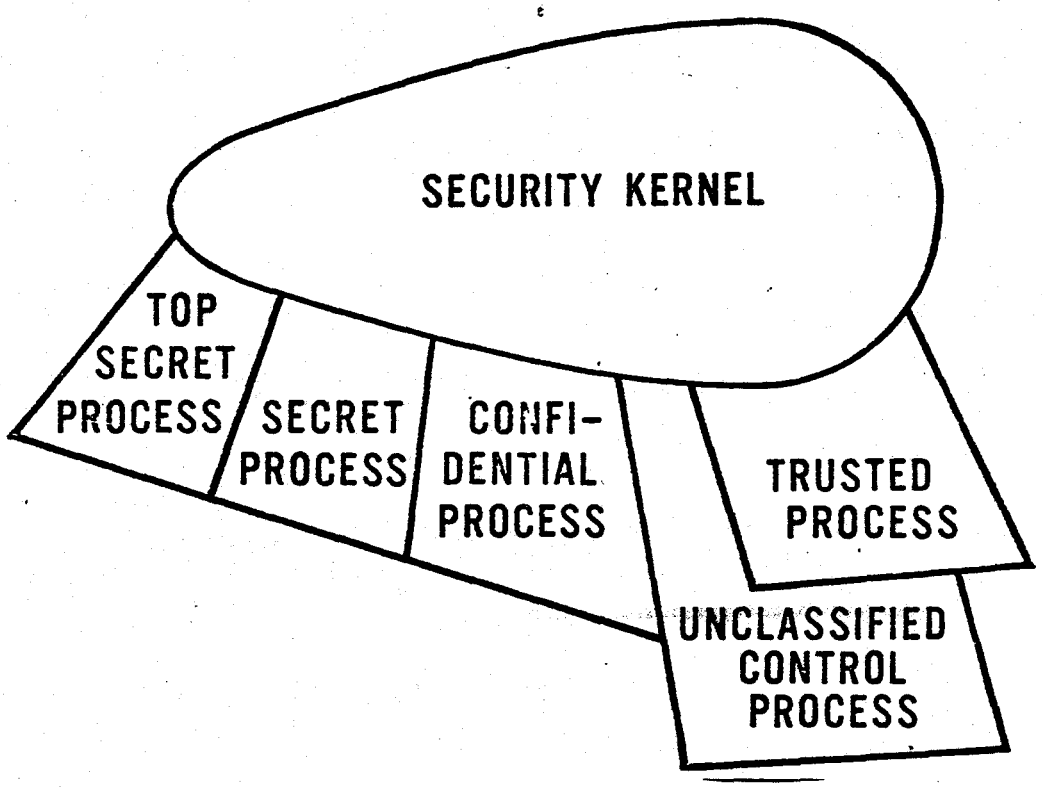
- MESSAGE RECEPTION & DISPLAY
- INTERNAL ANNOTATION & DISTRIBUTION
- ACTION ASSIGNMENT & LOGGING
- ON-LINE MESSAGE & NOTE FILING
- RETRIEVAL BY FILE, KEYWORDS, MESSAGE FIELDS
- MESSAGE & NOTE CREATION

  FROM SCRATCH; PREFORMATTED, PREADDRESSED, EXISTING INFO

- ON-LINE COORDINATION & RELEASE



ARCHITECTURE OF APPLICATION CODE



CONCLUSIONS

- STILL PROBLEMS TO BE SOLVED
- PAST DEVELOPMENTS HAVE SUFFERED FROM
  - LACK OF FULL-FUNCTION SECURE OS
  - LACK OF ADEQUATE HUMAN INTERFACE EVALUATION AND FEEDBACK
- FUTURE DEVELOPMENTS CAN AVOID THESE PROBLEMS

DOD COMPUTER SECURITY INITIATIVE

Stephen T. Walker
Chairman, DoD Computer Security
Technical Consortium

APPROVAL FOR DOD USE

DOD DIRECTIVE 5200.28

- ESTABLISHES DESIGNATED APPROVAL AUTHORITIES FOR EACH DOD COMPONENT FOR THE APPROVAL OF COMPUTER SYSTEMS PROCESSING CLASSIFIED INFORMATION
- INCLUDES PROVISIONS FOR THE APPROVAL OF MULTI-LEVEL SECURE SYSTEMS
- A FEW SYSTEMS HAVE BEEN APPROVED FOR MULTI-LEVEL USE
  - MULTICS - AF DATA CENTER

APPROVAL FOR DOD USE

- SEVERAL APPLICATIONS OF KSOS AND KVM ARE BEING DEVELOPED FOR USE IN SPECIFIC MULTI -LEVEL SECURE ENVIRONMENTS
- THESE APPLICATIONS ARE BEING COORDINATED WITH THE APPROPRIATE DESIGNATED APPROVING AUTHORITIES
- THIS APPROVAL PROCESS IS NOT VERY EFFICIENT AND CANNOT BE USED ON A WIDESPREAD BASIS

APPROVAL FOR DOD USE

- NEED CONSISTENT EVALUATION PROCESS APPLICABLE TO SYSTEMS DOD-WIDE
- NEED TO AVOID MULTIPLE EVALUATIONS OF THE SAME SYSTEMS FOR THE SAME APPLICATION

APPROVAL FOR USE

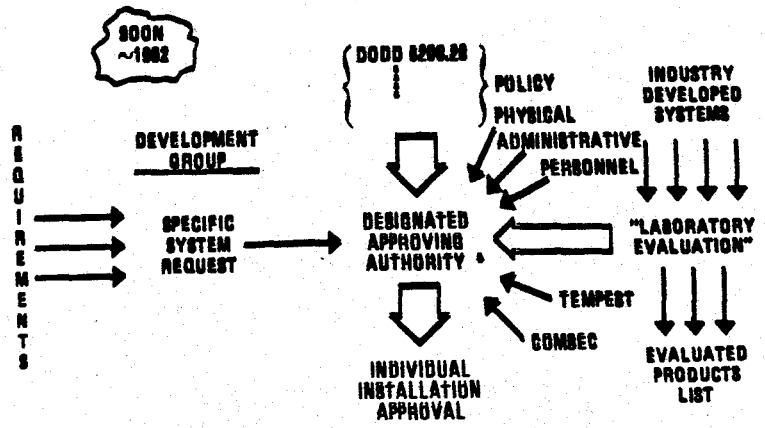
- ALL SECURITY RELATED DEVICES/TECHNIQUES HAVE VULNERABILITIES
- SECURITY MUST BE ACHIEVED BY COMBINATIONS OF MEASURES PROVIDED "IN DEPTH"
  - e.g., LOCKS DELAY INTRUDER ENABLING ROVING GUARD TO DETECT
- COMPUTER SYSTEMS WITH SIGNIFICANT INTEGRITY FEATURES, IN CONJUNCTION WITH OTHER PHYSICAL AND ADMINISTRATIVE SECURITY MEASURES, MAY BE SUITABLE FOR USE IN SENSITIVE MULTILEVEL ENVIRONMENTS
- TECHNICAL EVALUATION PROCESS WILL DETERMINE ENVIRONMENTS IN WHICH A PARTICULAR SYSTEM WILL BE SUITABLE, INCLUDING ADDITIONAL SECURITY MEASURES REQUIRED

APPROVAL FOR DOD USE

- NEED TWO-PHASE APPROVAL PROCESS
  - "LABORATORY APPROVAL" = DESIGN AND IMPLEMENTATION VERIFICATION
  - "SITE APPROVAL" = BY DESIGNATED APPROVAL AUTHORITY ACCORDING TO SITE SPECIFIC REQUIREMENTS AS RELATED TO "LABORATORY APPROVAL"

3548-8

APPROVAL FOR DOD USE



APPROVAL FOR DOD USE

- OSD HAS ESTABLISHED THE DOD COMPUTER SECURITY TECHNICAL CONSORTIUM TO
  - COORDINATE DOD COMPUTER SECURITY RESEARCH & DEVELOPMENT ACTIVITIES
  - PROVIDE TECHNICAL FOCUS FOR SYSTEM APPROVAL PROCESS
  - PROVIDE TECHNICAL FOCUS FOR INDUSTRY RELATIONS PROGRAM

COMPUTER SECURITY INITIATIVE