M.I.T. Laboratory for Computer Science
Computer Systems Research Division

February 25, 1977
Request for Comments No. 136

Information Security in a Distributed Computing System

Paul A. Karger

Attached is my recently accepted Master's thesis proposal.

Massachusetts Institute of Technology

Laboratory for Computer Science

Proposal for Thesis Research in Partial Fulfillment

of the Requirements for the Degree of

Master of Science

Title: Information Security in a Distributed Computing System

Submitted by: P. A. Karger
PSC Box 452
Hanscom AFB, Ma. 01731

Signature of Author

Date of Submission: 25 February 1977

Expected Date of Completion: 12 May 1977

Brief Statement of the Problem:

A key area in the protection of information in distributed computing systems is the interface between the host computer system and the rest of the network. While much work has been undertaken to provide communications security, primarily through encryption techniques, the mechanisms to assure information security in the host to network interfaces and the host to host interfaces are less well understood. This thesis research will examine various approaches for protection of information in distributed computing systems, identify those approaches that provide effective security, and clearly delineate those that, although they have been strongly supported in the literature, may only provide an illusion of security.

Supervision Agreement:

The program outlined in this proposal is adequate for a Master's thesis. The supplies and facilities required are available, and I am willing to supervise the research and evaluate the thesis report.

J. H. Saltzer
Professor of Computer Science and Engineering

Table of Contents

I Introduction

1.1 Growth of Distributed Systems

Distributed computing systems are becoming more and more common throughout the computing industry. Although we have had distributed systems of one sort or another since the development of the SAGE air defense system <Everett57> in the 1950's, the dramatic reductions in the cost of computing hardware have led to a growing feeling that distributed computing systems offer a number of advantages in providing efficient and economical computational power to the user.

1.2 Need for Information Security

The need for protection of information in computer based systems is clear. The numerous examples of computer related crimes <Parker73>, the need to protect national defense information, and the recent passage of laws guaranteeing the protection of personal data <Privacy74> have all led to a growing awareness and concern for computer security. Distributed systems can have an adverse impact on the security of an individual computer system by:

> "Potentially making the security controls on a specific host irrelevant by making information accessible to other hosts that do not have effective security controls," and by
>
> "Introducing additional vulnerabilities through the lack of effective security controls in network elements, e.g., insecure network communications processors." <Schell74>

However, a distributed computing system can also enhance the security of some computational tasks. Distribution of computing resources can introduce protection through physical separation, and a properly designed communications subsystem can ensure confinement of sensitive information within selected boundaries.

1.3 Confusion in the Literature

Just as the meaning of the term "distributed computing" has become very confused, meaning anything from time sharing with remote terminals to array processing to networks of independent cooperating systems, the topic of protection of information in a distributed computing system has become very confused in the literature. A variety of approaches have been proposed to achieve protection of information. Some proposals have been limited to specific special cases. Others have claimed global applicability. Some approaches proposed for protection in fact yield to relatively simple sorts of attacks. Review of the published literature presents little guidance to the distributed system builder as to which approaches work, which do not work, and when particular approaches should or should not be used.

1.4 Definitions

This section provides working definitions for the terms "distributed computing systems" and "protection of information." While global applicability cannot be claimed for these definitions, they appear accurate and suitable for a wide variety of potential applications.

1.4.1 Distributed Computing Systems

A distributed computing system consists of a collection of processing elements that are physically separated and use a communications type protocol. Thus, a time sharing system with simple teletype terminals is not a distributed computing system, nor is a tightly coupled multiprocessor system. (1) Distributed systems can be made up of a set of identical processors, all performing a single special purpose function, (2) or they may be made up of heterogeneous processors, each performing independent functions with only occasional inter-processor coordination. (3)

Enslow offers a more restricted definition of distributed computing systems by requiring them to meet the following criteria:

> "1) two or more general purpose processors (of any size: micro to maxi)

(1) Taking this definition to its extreme, a bus - oriented multiprocessor could be considered to be a distributed computing system. For purposes of this thesis research, such an extreme definition will be excluded.

(2) SAGE <Everett57> is an example of such a dedicated single function distributed system. Each SAGE center was capable of passing aircraft tracks to neighboring centers and correlating tracks of the same aircraft computed by two different centers. SAGE is certainly not a very interesting system from the point of view of research in the functionality of distributed systems except for a historical perspective. However, from a security and protection point of view, systems such as SAGE have many of the same characteristics as more sophisticated distributed systems.

(3) The ARPANET is an example of such a heterogeneous system.

"2) a system operating system

"3) the employment of a communications type protocol

"4) services are requested by name

"5) non-deterministic resource allocation." <Enslow76>

Enslow's definition excludes such systems as SAGE in which all resources are pre-allocated to dedicated tasks and such systems as the ARPANET in which there is no system operating system. (1) However, such excluded systems will be considered in this thesis research, because they share most of the same information protection requirements as systems that meet Enslow's definition.

1.4.2 Protection of Information

Protection of information means that information stored or manipulated in a computer system is not released to or modified by individuals (ie: human beings) who are not authorized to receive or modify that information. It is assumed that a high threat environment exists in which unauthorized, malicious individuals or organizations are willing to invest large amounts of money and to commit illegal acts to obtain illicit information. Such a high threat environment exists for national defense information and for high value civilian data such as electronic funds transfer or stock transfer information. Individuals who are authorized access to information are presumed

(1) Note that distributed systems meeting Enslow's definition have been implemented on the ARPANET. See sections 2.2 and 2.3 below.

to be trustworthy for the information to which they are authorized access. However, the software that trustworthy individuals run cannot, in general, be assumed to be trustworthy. A user's program may have been surreptitiously "bugged" to release information to other presumably unauthorized individuals. Such so-called "Trojan Horse" software (1) performs all its expected functions, but has undetected side effects that attempt to release information.

In addition to software threats, there are threats of wire tapping and physical theft of information storage media. It is assumed that physical securing will assure protection of information storage media (and for that matter, the front panel of the computers). However, physical security cannot be assumed at all nodes of a distributed computing system. Terminals and the newly developing "personal computers" may often reside in physically insecure locations. Wire tapping is a constant threat if communications links are not physically secure over their entire length. Since distributed systems may often choose to use either commercial insecure telephone lines or broadcast radio, wire tapping (or more appropriately for wireless communication - electronic eavesdropping) must be addressed to assure protection of information.

(1) The term "Trojan Horse" was first suggested by D. Edwards. <Anderson72>

1.5 Intent of this Research

It is the intent of this thesis research to examine the issues and requirements of security in distributed computing systems to develop a coherent approach to the protection of information consistent with intended uses. The overall plan of the thesis will be to examine specific scenarios of intended (or existing) distributed systems to identify mechanisms adequate to protect the information, while maintaining the necessary functionality to the users. Some protocols currently proposed in the literature may be rejected as demonstrably insecure. Others may prove unnecessarily restrictive. Examples will be drawn in part from the author's experience in system engineering of distributed systems for the United States Air Force Electronic Systems Division.

1.6 Exploit New Technologies

In pursuing this research, two new technologies - security kernels and very inexpensive LSI processors - seem to offer particular advantages in securing distributed computing systems. This thesis research will exploit these new technologies extensively.

1.6.1 Security Kernels

Numerous penetration studies have demonstrated that conventional computer systems do not have effective security controls. Such systems as GCOS <Anderson71>, IBM 360/370

<Abbott76>, TENEX <Abbott76>, and Multics <Karger74> have been examined and found lacking in effective security controls. This insecurity is a fundamental weakness of conventional systems and cannot be corrected by merely modifying or patching the operating systems. Even if every known security weakness in a particular system were repaired, there would be no basis to believe that every weakness had been found. Further, modifications required to repair the security vulnerabilities are often so complex that they themselves may introduce new security weaknesses. Anderson <Anderson72> reports that after extensive security "repairs" were made to one large commercial system that had been penetrated, a second penetration effort succeeded after less than one man-week of effort.

To overcome the weaknesses of conventional systems, an approach based on the use of security kernel was proposed <Schell73> to assure the effectiveness of the security controls of future systems. The security kernel of an operating system mediates all accesses to information, assuring that the desired security policies are enforced. To provide a practical basis for security, the kernel must meet the following engineering requirements:

a. Completeness: The security kernel must be invoked on every access to information.

b. Isolation: The security kernel must be protected from unauthorized tampering.

c. Certifiability: The security kernel must be small enough and simple enough that its correctness can be verified.

These engineering requirements in turn impose requirements on an actual implementation of a security kernel. To provide efficient complete mediation, descriptor based addressing (1) must be used on all references by the CPU to memory. (2) To meet the isolation requirement, the kernel must reside in the most protected state of a multiple protection state computer. (3) To meet the certifiability requirement, the security kernel must be designed from a formal model of secure processing <Bell75> and must be subjected to formal verification procedures <Millen76> to ensure correct implementation. Security kernels have been demonstrated for the PDP-11/45 by the MITRE Corp. <Schiller75> and by U.C.L.A. <Popek74> Kernels are presently under development for the UNIX <Lipner74>, and Multics <Schroeder75> operating systems and for the Honeywell Series 60 Level 6 <Honeywell76>. (4) In addition, the United States Air Force SATIN IV packet

(1) Descriptor based addressing is used in a variety of machines including the Honeywell Series 60 Level 68, the DEC PDP-11/45, the IBM 370, and the Burroughs 6700.

(2) Lampson's difficulties in implementing the CAL system <Lampson76> on a CDC6400 without descriptor based addressing hardware support this requirement.

(3) For example, the kernel would reside in kernel mode of the 3-state PDP-11/45, or in ring 0 of the 8-state Honeywell Series 60 Level 68 processor.

(4) Headquarters, United States Air Force Systems Command has directed that work on the Multics and the Series 60 Level 6 security kernels be terminated in 1977.

switched network is using portions of the security kernel technology in the Internal Access Control Mechanisms (IACM's) present in each SATIN IV communications processor.

This thesis research will assume that security kernel technology will be available for both large host computer systems and small communications systems. However, since many existing systems do not have security kernels, distributed systems will be examined in which some hosts are insecure, but in which secure communications processors can provide one-way communications in a secure fashion. (See section 3.3 below.)

1.6.2 LSI Technology

The development of LSI technology over the last few years has made possible today's very inexpensive microprocessor systems. A short term extrapolation of this technology leads one to expect small to medium scale computer systems (in the range of a present day PDP-11/70) to be available as single chip or single board systems for under $100. This type of technology dramatically expands the scope of distributed computing systems by making possible "personal" computing systems on a large scale. Such very inexpensive computers can simplify the protection problems by allowing physically separate computers to be dedicated to each category of information that must be protected. In addition, one can conceive of such a system resident in each user's display terminal. If the terminal system supported some type of security kernel, it might be possible to use this

"secure" terminal to access various "non-secure" systems without the danger of unauthorized leakage of information between systems. (See section 3.2 for a more complete discussion.) This thesis research will examine the possibilities of exploiting such dedicated LSI processors as an aid in resolving security problems.

II Review of Existing Approaches

This section is a review of several existing (or proposed in the literature) approaches to security in distributed computing systems. Problems and drawbacks of several of the approaches are identified. This thesis research will address these problems and attempt to propose solutions.

2.1 ARPANET TELNET and FTP Protocols (1)

The existing ARPANET TELNET and FTP protocols <Feinler76> provide very limited support for protection of information. (2) Each host is responsible for assuring its own protection, primarily by requiring password authentication at the time a network connection is made. Using these protocols, a user must remember different passwords (3) for each machine he or she uses and must transmit these passwords through various host machines, leaving large opportunities for the passwords to be stolen. Alternatively, the user could store passwords for the server

(1) TELNET is a protocol to provide remote terminal communication over the ARPANET. Using TELNET, a terminal connected either to a host processor or a terminal concentrator can communicate with any host on the network. FTP is a protocol to provide file transfer capabilities between hosts on the ARPANET.

(2) Of course the ARPANET Interface Message Processors (IMP's) were not developed to be "penetration-proof." While IMP software is quite reliable, it has not undergone the formal verification necessary to assure security. In addition, IMP-IMP communications lines are not encrypted. Therefore, the ARPANET is presently vulnerable to wire tapping. ARPA is presently sponsoring development of an end-to-end encryption device for the ARPANET called the Private Line Interface (PLI) <IMP76>.

(3) Users may often choose the same password for all sites, making password guessing easier.

machines in files on the user machines. This technique, however, extends the vulnerability of the passwords.

2.2 RSEXEC

RSEXEC (the Resource Sharing Executive for the ARPANET <Thomas73>) provides a much more sophisticated distributed environment than the TELNET and FTP protocols do. RSEXEC allows a user to view the file systems of several machines on the ARPANET as a single file system. (1) The user can name files at distant sites just as he or she can name local files. However, from an authentication point of view, RSEXEC is very similar to TELNET and FTP. Passwords must be stored on user machines and transmitted to server machines whenever network connections are made. Although RSEXEC hides much of the password processing from the user, the stored passwords for other systems remain subject to attack, either in the user system or while being transmitted through the network.

2.3 National Software Works

The National Software Works (NSW) <NSW76> is another distributed computing facility being implemented on the ARPANET. The NSW is intended to support software development activities by providing access to software development tools resident on various hosts. As in RSEXEC, the user of the NSW is sheltered

(1) Currently the RSEXEC protocols are only completely supported for the TENEX operating system. RSEXEC is partially supported by the ITS and Multics operating systems.

from the issues of where on the network particular files or tools are stored. Authentication is very simple in the NSW. If a user wishes to login to the NSW, he or she first connects to a local Front End (FE) process running on a local machine. The FE delivers the user's login request and authentication password to the Works Manager (WM), which runs on some centralized machine in the NSW. All requests for service must go through the FE to the WM. The WM then forwards service requests to appropriate systems. All systems must trust the WM implicitly. No authentication is performed, but rather any WM request must be honored without question. A host can assure itself that it is talking to the real WM by a scheme of dedicated network sockets, but all systems must accept the central authority of the WM.

2.4 Network Security Centers

One approach to security in distributed systems is the concept of a Network Security Center (NSC) proposed by Branstad <Branstad73, Branstad75> and expanded upon by Heinrich and Kaufman <Heinrich76>. The NSC is a centralized facility that, as proposed by Branstad, provides a secure cryptographic key distribution service. As such, the NSC can assure identification and authentication of the user to the servicing host computer and vice versa. The authentication is implicit in the cryptographic keys. Note that the function of the NSC is analogous to the function of the Works Manager (WM) in the National Software Works (NSW). While such a centralized authority may be acceptable to a network under a single management control, it may not be at all

acceptable in a distributed computing system that does not have central management. Diffie and Hellman <Diffie76> suggest an approach to avoid the necessity of trusting a single central authority. They suggest the use of multiple independent NSC's, each of which verifies and authenticates a requested two-way communication. Each NSC provides a cryptographic key to the sender and receiver processes. All the cryptographic keys are exclusive - ored together to produce a key not known to any of the NSC's involved.

Heinrich and Kaufman, however, attribute the NSC with security capabilities beyond those proposed by Branstad. In particular, they claim that the NSC can prevent unauthorized access to data by legitimate users of the network. However, the NSC is in fact unable to prevent such unauthorized accesses, unless the various host processors are themselves secure. The following two examples demonstrate the inability of the NSC to prevent unauthorized access.

For the first example, assume a network consisting of three host computers - A, B, and C. Assume A, B, and C do not have effective internal security controls, but that they communicate only via the network using cryptographic keys provided by the NSC. Assume A is authorized to access B's data base, and B is authorized to access C's data base, but A is not authorized to access C's data base. The NSC enforces the protection of C's data base by not providing a common cryptographic key to A and C. However, the NSC can do nothing if B forwards data from C to A.

The example above uses multiple systems connected to a network to leak information to unauthorized users. In fact, the NSC cannot prevent unauthorized access, even if there is only one system involved. Effectively, the NSC's granularity of protection is entire computer systems. If a user can gain access to a host system for some legitimate purpose, that user can bypass the nominal security controls of the host to access any information. Since such an attack would take place entirely within a single host, it would be invisible to the NSC.

Heinrich and Kaufman describe the NSC as maintaining an access matrix similar to Lampson's <Lampson71>. However, as shown by Harrison, et al. <Harrison76>, merely maintaining an access matrix does not guarantee that unauthorized access does not occur. In the first example, using Harrison's terminology, B can leak to A the generic right to read C's data base. In the second example, one can model the ineffective security controls of the operating system as a special subject in the access matrix. The special subject has access to all data in that particular system, but due to the ineffective security controls, all other subjects on that system have access to the special subject. Thus, if the NSC grants a user access to that particular system, by transitive closure on the access matrix, the user has been granted access to all data stored in the system. While Heinrich and Kaufman imply that the host system might choose to add additional protection, they do not make clear

that unless the host itself maintains effective security controls, the NSC can leave significant vulnerabilities open. This issue is of paramount importance if one wishes to build secure distributed systems in which some of the component host systems are fundamentally incapable of providing effective access controls.

2.5 Military Networks

The U.S. Department of Defense is presently developing two packet switched networks - SATIN IV for the U.S. Air Force Strategic Air Command and AUTODIN II for joint service communications. These networks achieve protection of classified message traffic by encrypting communications on a link by link basis and providing effective security controls in each message processor. For example in SATIN IV, each message is labelled with a security level and category set and the Internal Access Control Mechanism (IACM) of each communications processor assures that messages are routed only to destinations that are properly cleared to receive them. When messages are entered into SATIN IV from an external interfaced system, the IACM must differentiate between interfaced systems with effective security controls and interfaced systems without effective security controls. Systems without effective controls cannot be trusted to properly label messages. Messages from such untrusted systems must be treated by SATIN IV as classified at the highest level processed by that

particular system. (1) The IACM's of each communications processor will be verified to operate correctly and provide effective security controls. SATIN IV does not provide a virtual circuit between host systems. Rather it provides a datagram <Pouzin76> type of service by just delivering messages. As such, SATIN IV could implement the type of one-way communication hypothesized below in section 3.2. The security characteristics of AUTODIN II are less well defined at this time, but are expected to be similar to SATIN IV. A brief summary of the requirements of both the SATIN IV and AUTODIN II systems can be found in <Chandersekaran76>.

2.6 Dynamic Process Renaming

Farber and Larsen <Farber75> suggest an approach to security in a ring network by dynamically renaming the process names that appear in the message destination fields. They reason that if in a series of messages sent from one process on a host to another process on a different host, the destination fields of the messages are changed in every message based on presumably secret transformations known only to the source and destination systems, then an intruder could not follow the rapid exchange of messages and would be unable to extract information. Farber and Larsen describe synchronization and error detection methods that make

(1) When presented with a message labelled at a lower level than the highest level processed by the untrusted message source, the IACM must either generate a security alarm or relabel the message at the highest level processed by that system. Operational requirements will determine which is appropriate.

dynamic process renaming a practical communications protocol. Unfortunately, they do not address the possibility of computer assisted traffic analysis which could easily distinguish patterns in the traffic and determine the content of the transmissions. For example, the login dialog to Multics <Honeywell75> is a very stylized one in which Multics sends a greeting message, the user responds with a login command, Multics requests the user's password, and the user types it in. Such a dialog could easily be recognized in a recording containing many unrelated messages. Such an analysis was done in one penetration of a computer system, documented in <Computerworld75>, in which the penetrator examined teletype communications buffers to collect passwords of many users. An even easier example would be traffic generated by a program like the MACSYMA system <MACSYMA75> in which messages from the program to the user are sequentially numbered so that the user can reference them easily in later requests. Clearly, dynamic process renaming is effective only against very unsophisticated attacks. Encryption of communications is much more effective against sophisticated penetration attempts.

2.7 Encryption

As alluded to above, encryption forms a major part of security in any sort of data communications system. Unless the communications medium can be physically guarded over its entire length, there is an ever present threat of wire tapping to either extract information or introduce bogus traffic. Two basic types of encryption can be used: link encryption and end-to-end

encryption. In link encryption, every communications link is equipped with a pair of encryption devices. In end-to-end encryption, messages are encrypted at their source and not decrypted until they reach their destination. The use of encryption is discussed in <Kent76> and <Diffie76> and will not be covered here. Throughout this thesis, it will be assumed that all communications are encrypted, either with link or end-to-end techniques.

2.8 External Security Monitors

Painter <Painter75> proposes an approach to security in computer networks in which an external minicomputer is attached to each host processor to monitor all hardware and software operations for security malfunctions. To monitor the hardware, Painter proposes equipment analogous to Automatic Test Equipment (ATE) be attached to run periodic tests on all hardware components. Since hardware can fail randomly, such tests are important for the operation of any secure computer facility. Software versions of hardware security monitors have been implemented for the Honeywell 645 <Karger74> and for the Honeywell 6180 <Hennigan76>. Painter points out two major difficulties with his external hardware monitor proposal. First, ATE normally interferes with hardware performing its normal operational functions. Therefore, either ATE must be designed that does not interfere, or redundant hardware must be provided to allow checking of components on an offline basis. In the latter case, software must also exist to allow reconfiguration of

the hardware without disruption of ongoing processing. Second, as LSI technology advances, it becomes more and more difficult to build ATE. Because of the concentration of functions on single chips, it becomes impossible to break systems down into separate "black boxes" for isolated testing. Perhaps future LSI hardware can be designed with additional leads for ATE interfaces. Painter also points out that his technique cannot detect Trojan Horses that may be concealed in LSI chip designs.

Unfortunately, Painter's scheme for external software monitors is less well founded than his hardware monitor scheme, because software, unlike hardware, does not fail randomly. Software security systems are either correct or incorrect from the start. They do not "fail" after a period of time. Painter proposes that software security surveillance be carried out by hardware performance evaluation monitors that examine the contents of registers and main memory "looking" for security penetrations. Painter admits the hopelessness of analyzing every operation performed by the host computer. The CPU time required would be many times that required by the host computation itself. Painter instead suggests a statistical approach, periodically checking the host for software security penetrations. However, the types of penetrations described in <Karger74> can be consummated in a matter of microseconds. The probability of statistically discovering a well rehearsed penetration is extremely small. More importantly, Painter offers no evidence that such an external software security monitor can be

effectively implemented, even given unlimited CPU time. Essentially, Painter expects the monitor to examine arbitrary programs running in the host system to see if they ever enter an insecure state. One can draw an analogy to Turing Machine theory which shows the undecidability of the question of whether an arbitrary Turing Machine ever enters a particular state. While a proof that Painter's approach is not effectively computable is beyond the scope of this thesis proposal, the feasibility of his approach is certainly open to question.

Painter takes his software surveillance monitor concept one step further and suggests that the monitor could be implemented on the host system itself. This technique of self-monitoring is shown to be insecure in <Karger74>. If the host system is secure, then the self-monitor could be useful in detecting some, but not necessarily all, unsuccessful penetration attempts. However, if the host system is not secure, then the successful penetrator will immediately modify the self-monitor to assure that it only reports that all is well.

III Thesis Objectives and Approach

The primary objective of this thesis research is to identify the appropriate mechanisms for protection of information in distributed computing systems. The basic approach will be to examine a number of contemplated or actual distributed computing systems in which there are information protection requirements. In each case, the protection issues will be clearly defined, separating inherent technological issues from detailed requirements of particular applications. It is not expected that this thesis will identify major new protection mechanisms. Rather, it will examine and clarify what we mean by protection in distributed systems, identify suitable mechanisms for protection in distributed systems, and clearly delineate mechanisms that provide only an illusion of security.

In examining the various security mechanisms for distributed computing systems, it is clear that there are two major classes of issues that must be addressed - authentication and enforcement of access rights. Much of the confusion in the literature seems to come from mixing these issues together. (See, for example, section 2.4.) Part of the confusion arises from the fact that the two issues cannot be treated entirely separately. The interface of authentication to the rest of the security system is crucial to maintaining effective security. This interface will be examined and specified for distributed systems as part of this thesis research.

The remainder of this section first outlines the primary semantic models of access control to be considered. It then examines two particular security mechanisms in some depth. The first is an authentication mechanism, and the second is a mechanism for enforcing access rights in a lattice-type security system. (Lattice-type security systems are discussed in section 3.1.)

3.1 Semantics of Distributed Access Control

The most general model of access control is Lampson's access matrix <Lampson71> in which the access rights of each subject to each information containing object are defined in entries of the matrix. Normally, subjects are represented by rows of the matrix and objects by the columns. By introducing attributes such as "owner" or "control", the matrix can define not only access rights to objects, but also access rights to change entries in the access matrix itself.

Two generic implementations of the access matrix have evolved that encompass most actual computer security systems. Treating the access matrix by columns, we get an Access Control List (ACL) system such as is used in Multics <Organick72>. Each object has an associated ACL which lists the access rights of subjects. When a subject wishes to gain access to an object, the ACL must be consulted to determine access rights.

If the access matrix is treated by rows instead of columns, we get a capability system such as is described by Fabry

<Fabry74>. Each subject has a capability list, which describes the objects to which the subject has access rights. When a subject wishes to access an object, the subject merely invokes the appropriate capability. Possession of the capability implies that the subject has access rights to the object.

One can imagine implementations of either ACL or capability systems in a distributed environment. Section 3.2 below discusses forwarding of authentication information for use in a distributed ACL system. One could also imagine a distributed system that passed capabilities from computer to computer. The capabilities might have to be encrypted to prevent forgery, and certain problems such as garbage collection would remain unresolved. However, such a distributed capability system could certainly be hypothesized.

Unfortunately Harrison <Harrison76> has shown that certain access control properties are undecidable for fully general access matrices. In particular, the so-called "confinement problem" is one such undecidable property. The "confinement problem," identified by Lampson in <Lampson73>, is a requirement that a subject who has access to an object cannot leak the information contained in that object to some other subject not authorized the information. The "confinement problem" assumes that the subject in question may be executing programs containing "Trojan Horses." (See section 1.4.2.)

Lipner <Lipner75> and Denning <Denning76> have shown that lattice-type security systems can be implemented to solve portions of the "confinement problem." The lattice model of security, initially formally specified by Bell and LaPadula <Bell75> and Walter, et al. <Walter75>, assigns a security "classification" to each object and a security "clearance" to each subject. The classifications and clearances are drawn from a partially ordered set of security "levels." For a subject to access an object, it must be "cleared" for that object. More formally, if a subject S wishes to read an object O, the classification of O must be less that or equal to the clearance of S. To assure confinement, two alternative policies may be enforced. Bell and LaPadula <Bell75> define the "*-property", which requires that if a subject S wishes to write into an object O, the clearance of S must be less than or equal to the classification of O. Alternatively, Weissman's ADEPT-50 system <Weissman69> enforced a "high water mark" rule in which if a subject S wished to write into an object O, the classification of O must be upgraded to the current clearance of S. (1)

While the lattice system is a significant restriction on the general access matrix, it is the only useful system that has been proposed to date for which the "confinement problem" is

(1) Two points must be noted. First, as Denning <Denning76> points out, ADEPT-50 did not precisely implement this definition of "high water mark," but in fact had some "bugs" relating to confinement. Second, because the security "levels" form a partial ordering, S's clearance may be neither greater than nor less than or equal to O's classification. See <Bell75> for further discussion of disjoint levels.

decidable. In addition, the lattice model seems to adequately represent many of the security requirements faced by "real" distributed (and non-distributed) computing systems. The lattice model directly represents military security systems and a wide variety of corporate security systems for protection of classified and proprietary information respectively. Turn <Turn76> describes several proposed privacy protection schemes that are variants of the lattice model.

3.2 Forwarded Authentication by Proxy Login

One possible solution to the problem described in sections 2.1 and 2.2 of storing passwords (1) on various systems throughout a network is to extend Montgomery's <Montgomery76> concept of forwarded authentication to distributed systems. This extension can be achieved by noting that forwarded authentications are in fact a generalization of the proxy login concept that has been proposed for Multics <Saltzer74> but has never been implemented. In proxy logins, a user could allow another user to proxy for him without releasing his password. The second user would request a proxy login, give his own password, and, assuming the first user had granted permission, the second user would be logged in under the first user's ID. Such a concept could be extended to a distributed environment in

(1) Although passwords are discussed here as the authentication medium, any other form of authentication could serve equally well. Authentication could be based on magnetic stripe credit-type cards, or fingerprint readers or other more sophisticated authenticators. Cotton and Meissner <Cotton75> and Richardson and Potter <Richardson73> describe several sophisticated authentication schemes.

which a user Jones on host A could declare that any login requests from user M4473 on host B were to be honored as proxies. In this way, Jones' password need not be stored on system B, and system A need not grant unlimited trust to system B. The M4473 user has no more access on system A than user Jones had, thus limiting potential damage. Two problem areas in this approach will require further study. First, Jones requires a way that is not vulnerable to Trojan Horse attack to specify who may proxy login. Second, by granting proxy login to M4473, Jones has implicitly granted proxy login to the system administrator of system B, to users of system C to whom M4473 may have granted proxy login, and to any users who can penetrate system B's security controls. While this is not strictly a security problem, since Jones presumably knows what he is doing, it is an issue in human engineering of the security controls to ensure that users are not "surprised" by what they do.

3.3 Secure Networks of Insecure Hosts

3.3.1 One Way Communication

As discussed in section 1.6.1, many present day computers cannot support effective security controls. <Smith75> However, due to the large capital investments in existing systems, there is a large demand to build distributed systems out of insecure components. Using secure communications processors such as those in SATIN IV (See section 2.5.), one could build a network of insecure systems that could enforce a lattice security system in

which a partial ordering can be defined among the various security "levels." As an example, consider a system with two security levels - sensitive and non-sensitive. If one dedicates processors within the distributed system to particular security levels, the secure communications processors can assure confinement of highly sensitive information by providing one way communications from non-sensitive systems to sensitive systems. Now, no matter what the software processing sensitive information attempts to do, it cannot leak sensitive information to a non-sensitive system on the network. Indeed it cannot leak any information at all. The communications must be truly one way, because, for example, a Trojan Horse in the sensitive system could communicate classified information out encoded in the number of message acknowledgments returned. Two major questions come out of the concept of one way communication. First, what types of applications can function with one-way communication? Clearly many closely coupled applications are impossible, but certain types of transaction oriented systems should be feasible. For example, a corporation's day to day operations might be considered non-sensitive. However, corporate planning functions which must read the day to day transaction data base are considered sensitive. If two copies of the data base were maintained, one for day to day operations and one for corporate planning, one-way communication could be used to update the sensitive corporate planning data base automatically. Second, how can reliable communications be achieved without message acknowledgment? Preliminary examination of this question

indicates that the communications processors could assume responsibility for storing and retransmitting messages if errors occur. By using this technique, the destination host need not acknowledge messages to the source, but only to the secure network.

3.3.2 Semi-Automated Downgrading

In section 3.2.1 above, we described the limitations of one-way communications systems. Stork <Stork75> describes a technique for downgrading sensitive information based on a formulary approach that could allow some limited forms of reverse communications. For example, if one could write a program that could examine messages to determine whether they contained sensitive information, that program (called a formulary by Hoffman <Hoffman70>) could mediate all reverse communications, ruling on the propriety of each message going from the sensitive system to the non-sensitive system. The formulary program could be implemented on a security kernel based processor interposed between the sensitive (but not trustworthy) system and the non-sensitive system.

Formularies may be very difficult to implement. They may even require natural language understanding capabilities that are beyond the current state of the art. As a result, one could make do with a very simple formulary that displayed each message for human review, and then securely implemented the human's decision. This use of human review leads to the title of this

section, "Semi-Automated Downgrading." Human review introduces obvious limitations in the bandwidth of any communications, but may be acceptable for some applications. For example, the MITRE Corp. has implemented a security kernel demonstration <Mack76> that includes semi-automation of human review for downgrading. In the demonstration scenario, an intelligence officer in an air defense center selectively downgrades classified air tracks, so that an uncleared operations officer may see upcoming threats.

Even human review for downgrading of messages must be strictly limited. A Trojan Horse program on the sensitive system could attempt to fool the human reviewer by encoding sensitive information in non-printing characters between legitimate words of a message. Since the non-printing characters are not displayed, the human reviewer would be unaware of their presence. This type of attack is described in <Nibaldi76>.

3.3.3 Secure Intelligent Terminals

Another approach to reducing the limitations of a one-way communications system is to provide an intelligent secure terminal. Such a terminal would include a microprocessor with a very simple security kernel that allowed the user to communicate simultaneously with two or more host systems operating at different dedicated security levels. The user could then issue commands to low level systems based on results derived from high level systems. Human engineering is of paramount importance to assure that the user does not accidentally mix different levels

of information, particularly if a Trojan Horse program on the high level system is attempting to fool the user. (1)

(1) Note that the user must be presumed trustworthy although his programs may not be. There is nothing a computer system can do if a cleared user is untrustworthy. The computer system must rely on external personnel clearance systems to determine who is or is not trustworthy. Trustworthiness, however, does not imply infallibility. The computer system must try to protect the user from being fooled or making an inadvertent mistake.

IV Schedule and Resource Requirements

4.1 Schedule

The following schedule is proposed for this thesis research:

January 1977 - begin literature search
prepare thesis proposal

February 1977- thesis proposal approved
complete literature search
begin developing paradigms

March 1977 - complete paradigm specifications
write first draft of thesis

April 1977 - review paradigms
revise thesis as required

12 May 1977 - complete thesis

4.2 Resource Requirements

Computer time on the MIT Information Processing Center's Multics system may be required for small experiments with some of the proposed protocols. Depending on the exact nature of any required experiments, computer time may also be needed on the Multics development system at Honeywell's Cambridge Information Systems Laboratory.

References

<Abbott76> Abbott, R. P., et al., "Security Analysis and Enhancements of Computer Operating Systems," The RISOS Project, Lawrence Livermore Laboratory, Livermore, Ca., NBSIR 76-1041, National Bureau of Standards, Washington, D.C., April 1976.

<Anderson71> Anderson, James P., "AF/ACS Computer Security Controls Study," James P. Anderson and Co., ESD-TR-71-395, HQ Electronic Systems Division, Hanscom AFB, Ma., November 1971. (AD 251865L)

<Anderson72> Anderson, James P., "Computer Security Technology Planning Study," James P. Anderson and Co., ESD-TR-73-51, Vols. I and II, HQ Electronic Systems Division, Hanscom AFB, Ma., October 1972. (AD 758206 and AD 772806)

<Bell75> Bell, D. E. and L. J. LaPadula, "Computer Security Model: Unified Exposition and Multics Interpretation," The MITRE Corp., ESD-TR-75-306, HQ Electronic Systems Division, Hanscom AFB, Ma., June 1975. (AD A023588)

<Branstad73> Branstad, D., "Security Aspects of Computer Networks," AIAA Conference Proceedings, Huntsville, Al., April 1973.

<Branstad75> Branstad, D., "Encryption Protection in Computer Data Communications," Fourth Data Communications Symposium, Quebec City, Canada, 7-9 October 1975, pp. 8-1 - 8-7.

<Chandersekaran76> Chandersekaran, C. S. and K. S. Shankar, "Towards Formally Specifying Communication Switches," Trends and Applications 1976: Computer Networks, Institute of Electrical and Electronics Engineers, Inc., New York, N. Y., 17 November 1976, pp. 104-112.

<Computerworld75> "T/S Service Security Cracked by Schoolboy With Series of Tricks," Computerworld, Vol. IX, No. 5, 29 January 1975.

<Cotton75> Cotton, I. W. and P. Meissner, "Approaches to Controlling Personal Access to Computer Terminals," Proceedings of the 1975 Symposium on Computer Networks: Trends and Applications, Institute of Electrical and Electronics Engineers, New York, N. Y., 1975.

<Denning76> Denning, D. E., "A Lattice Model of Secure Information Flow," Communications of the ACM, Vol. 19, No. 5, May 1976, pp. 236-243.

<Diffie76> Diffie, W. and M. E. Hellman, "Multiuser Cryptographic Techniques," 1976 National Computer Conference, AFIPS Conference Proceedings, Vol. 45, AFIPS Press, Montvale, N. J., 1976, pp. 109-112.

<Enslow76> Enslow, P., "What Does Distributed Processing Mean?", "Distributed Processing Workshop Transcript," Computer Architecture News, Vol. 5, No. 5, December 1976, pp. 11-12.

<Everett57> Everett, R. R., C. A. Zraket, and H. D. Bennington, "SAGE - A Data-Processing System for Air Defense," Proceedings of the Eastern Joint Computer Conference, December 9-13, 1957, Washington, D. C., The Institute of Radio Engineers, Inc., New York, N. Y., 1958.

<Fabry74> Fabry, R. S., "Capability-Based Addressing," Communications of the ACM, Vol. 17, No. 7, July 1974, pp. 403-412.

<Farber75> Farber, D. J. and K. C. Larson, "Network Security Via Dynamic Process Renaming," Fourth Data Communications Symposium, Quebec City, Canada, 7-9 October 1975, pp. 8-13 - 8-18.

<Feinler76> Feinler, E. and J. Postel, ARPANET Protocol Handbook, NIC 7104, Network Information Center, Stanford Research Institute, Menlo Park, Ca., April 1976.

<Harrison76> Harrison, M. A., W. L. Ruzzo, and J. D. Ullman, "Protection in Operating Systems," Communications of the ACM, Vol. 19, No. 8, August 1976, pp. 461-471.

<Heinrich76> Heinrich, F. R. and D. J. Kaufman, "A Centralized Approach to Computer Network Security," 1976 National Computer Conference, AFIPS Conference Proceedings, Vol. 45, AFIPS Press, Montvale, N. J., June 1976, pp. 85-90.

<Hennigan76> Hennigan, K. B., "Hardware Subverter for the Honeywell 6180," The MITRE Corp., ESD-TR-76-352, HQ Electronic Systems Division, Hanscom AFB, Ma., December 1976.

<Hoffman70> Hoffman, L. J., The Formulary Model for Access Control and Privacy in Computer Systems, dissertation, Stanford Linear Accelerator Center, May 1970.

<Honeywell75> Honeywell Information Systems, Inc., Multics Programmers' Manual Commands and Active Functions, Order No. AG92, Rev. 1, January 1975, Section IV.

<Honeywell76> Honeywell Information Systems, Inc., "Security Kernel Specification for a Secure Communications Processor," ESD-TR-76-359, HQ Electronic Systems Division, Hanscom AFB, Ma., in progress.

<IMP76> Specifications for the Interconnection of a Host and an IMP, Report No. 1822, Bolt Beranek and Newman, Inc., January 1976, Appendix H.

<Karger74> Karger, P. A. and R. R. Schell, "Multics Security Evaluation: Vulnerability Analysis," ESD-TR-74-193, Vol. II, HQ Electronic Systems Division, Hanscom AFB, Ma., June 1974. (AD A001120)

<Kent76> Kent, S. T., Encryption - Based Protection Protocols for Interactive User - Computer Communication, SM thesis, MIT Dept. of Electrical Engineering and Computer Science, June 1976. (Also available as MIT/LCS/TR-162, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Ma., June 1976.)

<Lampson71> Lampson, B. W., "Protection," Proceedings Fifth Princeton Conference on Information Sciences and Systems, Princeton University, March 1971, pp. 437-443, reprinted in Operating Systems Review, Vol. 8, No. 1, January 1974, pp. 18-24.

<Lampson73> Lampson, B. W., "A Note on the Confinement Problem," Communications of the ACM, Vol. 16, No. 10, October 1973, pp. 613-615.

<Lampson76> Lampson, B. W. and H. E. Sturgis, "Reflections on an Operating System Design," Communications of the ACM, Vol. 19, No. 5, May 1976, pp. 251-265.

<Lipner74> Lipner, S. B., S. R. Beckhardt, and D. F. Stork, "A UNIX Executive for Use with the PDP-11/45 Security Kernel," WP-20056, The MITRE Corp., Bedford, Ma., 5 December 1974.

<Lipner75> Lipner, S. B., "A Comment on the Confinement Problem," Proceedings of the Fifth Symposium on Operating System Principles, ACM Operating Systems Review, Vol. 9, No. 5, November 1975, pp. 192-196.

<Mack76> Mack, J. L. and B. N. Wagner, "Secure Multilevel Data Base System: Demonstration Scenarios," The MITRE Corp., ESD-TR-76-158, HQ Electronic Systems Division, Hanscom AFB, Ma., October 1976.

<MACSYMA75> MACSYMA Reference Manual, Project MAC, Massachusetts Institute of Technology, Cambridge, Ma., November 1975.

<Millen76> Millen, J. K., "Security Kernel Validation in Practice," Communications of the ACM, Vol. 19, No. 5, May 1976, pp. 243-250.

<Montgomery76> Montgomery, W. A., A Secure and Flexible Model of Process Initiation for a Computer Utility, SM and EE thesis, MIT Dept. of Electrical Engineering and Computer Science, June 1976. (Also available as (TR-163, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Ma., June 1976.)

<Nibaldi76> Nibaldi, G. and B. Wagner, "Secure Multilevel Data Base Systems: System Software," MTR-3160, Vol. III, The MITRE Corp., Bedford, Ma., in progress.

<NSW76> Semi-Annual Technical Report, CADD-7603-0411, Massachusetts Computer Associates, Wakefield, Ma., 4 March 1976.

<Organick72> Organick, E. I., The Multics System: An Examination of its Structure, MIT Press, Cambridge, Ma., 1972.

<Painter75> Painter, J. A., "A Minicomputer Network to Enhance Computer Security," Communications Networks, Online Conferences, Ltd., Uxbridge, England, 1975.

<Parker73> Parker, D. B., S. Nycum, and S. S. Oura, Computer Abuse, Stanford Research Institute, Menlo Park, Ca., November 1973.

<Popek74> Popek, G. J. and C. S. Kline, "Verifiable Secure Operating System Software," 1974 National Computer Conference, AFIPS Conference Proceedings, Vol. 43, AFIPS Press, Montvale, N. J., 1974, pp. 135-142.

<Pouzin76> Pouzin, L., "Virtual Circuits vs. Datagrams - Technical and Political Problems," 1976 National Computer Conference, AFIPS Conference Proceedings, Vol. 45, AFIPS Press, Montvale, N. J., 1976, pp. 483-494.

<Privacy74> Privacy Act of 1974, Title 5, United States Code, Section 552a (Public Law 93-579), December 31, 1974.

<Richardson73> Richardson, M. H. and J. V. Potter, "Design of a Magnetic Card Modifiable Credential System Demonstration," MCI-73-3, HQ Electronic Systems Division, Hanscom AFB, Ma., December 1973.

<Saltzer74> Saltzer, J., "Protection and the Control of Information Sharing in Multics," Communications of the ACM, Vol. 17, No. 7, July 1974, pp. 388-402.

<Schell73> Schell, R. R., P. J. Downey, and G. J. Popek, "Preliminary Notes on the Design of Secure Military Computer Systems," MCI-73-1, HQ Electronic Systems Division, Hanscom AFB, Ma., January 1973.

<Schell74> Schell R. R. and P. A. Karger, "Security in ADP Teleprocessing Systems - ADP Host Role," presentation at the Electronics and Aerospace Systems Convention, Washington, D. C., October 7-9, 1974.

<Schiller75> Schiller, W. L., "The Design and Specification of a Security Kernel for the PDP-11/45," The MITRE Corp., ESD-TR-75-69, HQ Electronic Systems Division, Hanscom AFB, Ma., May 1975. (AD A011712)

<Schroeder75> Schroeder, M. D., "Engineering a Security Kernel for Multics," Proceedings of the Fifth Symposium on Operating System Principles, ACM Operating Systems Review, Vol. 9, No. 5, November 1975, pp. 25-32.

<Smith75> Smith, L., "Architectures for Secure Computing Systems," The MITRE Corp., ESD-TR-75-51, HQ Electronic Systems Division, Hanscom AFB, Ma., April 1975. (AD A009221)

<Stork75> Stork, D. F., "Downgrading in a Secure Multilevel Computer System: The Formulary Concept," The MITRE Corp., ESD-TR-75-62, HQ Electronic Systems Division, Hanscom AFB, Ma., May 1975. (AD A011696)

<Thomas73> Thomas, R. H., "A Resource Sharing Executive for the ARPANET," 1973 National Computer Conference and Exposition, AFIPS Conference Proceedings, Vol. 42, AFIPS Press, Montvale, N. J., 1973, pp. 155-163.

<Turn76> Turn, R., "Classification of Personal Information for Privacy Protection Purposes," 1976 National Computer Conference, AFIPS Conference Proceedings, Vol. 45, AFIPS Press, Montvale, N. J., 1976, pp. 301-307.

<Walter75> Walter, K. G., et al., "Initial Structured Specifications for an Uncompromisable Computer Security System," Case Western Reserve University, ESD-TR-75-82, HQ Electronic Systems Division, Hanscom AFB, Ma., July 1975.

<Weissman69> Weissman, C., "Security Controls in the ADEPT-50 Time Sharing System," 1969 Fall Joint Computer Conference, AFIPS Conference Proceedings, Vol. 35, AFIPS Press, Montvale, N. J., 1969, pp. 119-133.