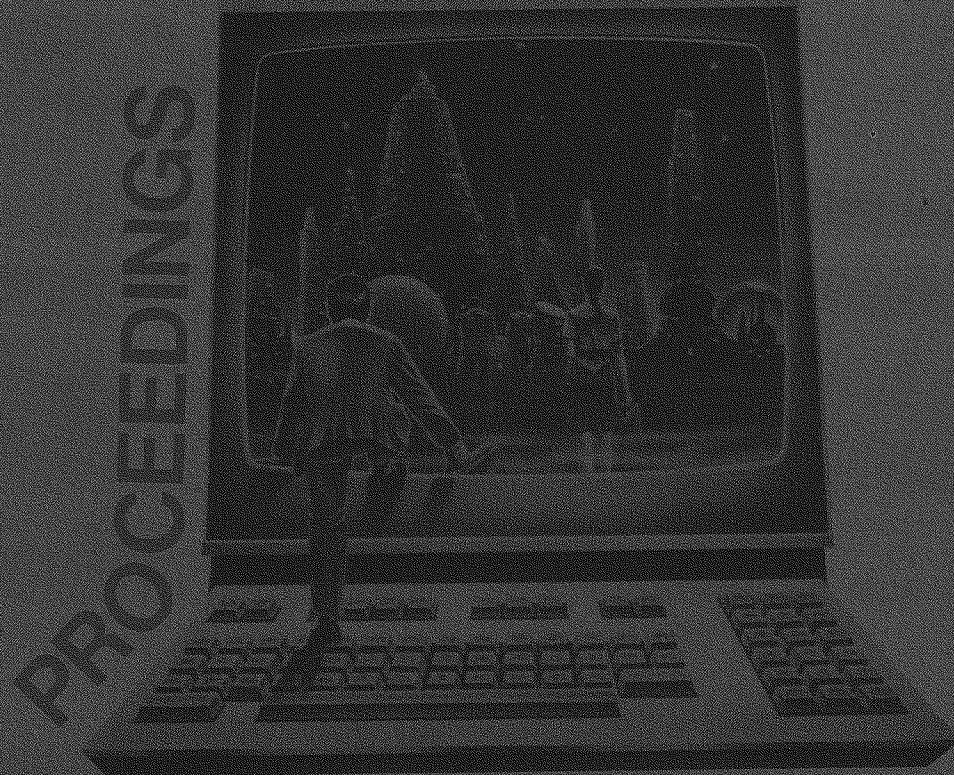


11th

National Bureau of Standards/National Computer Security Center

National Computer Security Conference



PROCEEDINGS

17-20 October 1988

RECEIVED

OCT 2 1989

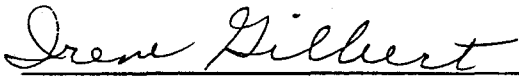
A POSTSCRIPT

“COMPUTER SECURITY...
Into The Future”

INTRODUCTION

This Postscript completes the official record of the Eleventh National Computer Security (NCS) Conference. As in past years, this Postscript contains several papers presented during the previous NCS Conference that arrived too late to be included in the Proceedings. In addition, the Postscript contains photographs of our award recipients, information on ordering IBM security awareness tapes that were discussed at the Speak Out session, and the questions and answers presented during the opening plenary session of the Eleventh Conference as well as those submitted during the session, most of which the panelists did not address for lack of time.

The plenary session questions and answers represent either joint responses agreed to by both the National Computer Security Center (NCSC) and the National Institute for Standards and Technology (NIST) or, where indicated, the views of one or the other of these organizations. This is the first complete and authorized exposition of the NCSC's and NIST's positions in light of the Computer Security Act of 1987. In that respect, it is a landmark document. It reflects the answers to difficult questions as of October 1988 and, therefore, represents an important historical perspective on U.S. computer security. We hope that you will find it enlightening and helpful.


IRENE GILBERT
NATIONAL INSTITUTE OF
STANDARDS AND TECHNOLOGY


ELIOT SOHMER
NATIONAL COMPUTER
SECURITY CENTER

TABLE OF CONTENTS

Page	
1	<i>A Secure DBMS Design</i> Thomas H. Hinke, Cristi Garvey, Nancy Jensen, Jackson Wilson, Amy Wu, TRW Defense Systems Group
14	<i>Evolution of a Model for Computer Integrity</i> David D. Clark, MIT Laboratory for Computer Science; David R. Wilson, Information Systems Consulting Services, Ernst & Whinney
28	<i>A Guide to Auditing for Controls and Security: A Systems Development Life Cycle Approach</i> Zella G. Ruthberg, National Institute of Standards and Technology
71	<i>The LOCK Demonstration</i> W. E. Boebert, Honeywell Secure Computing Technology Center
89	<i>Constructing an INFOSEC System Using LOCK Technology</i> W.E. Boebert, Honeywell Secure Computing Technology Center
96	<i>Update Processing in LDV: A Secure Database System</i> Paul Stachour, Bhavani Thuraisingham, Patricia Dwyer, Honeywell, Inc.
117	Ordering Information for IBM Security Films
119	Eleventh National Computer Security Conference Plenary Session Questions and Answers
175	National Computer Security Center EPL Certificates
177	National Institute of Standards and Technology Certificates
178	Outstanding Paper Awards

A1 Secure DBMS Design

Thomas H. Hinke, Cristi Garvey,
Nancy Jensen, Jackson Wilson, Amy Wu
TRW Defense Systems Group
One Space Park, Redondo Beach, Calif. 90278

Abstract

This paper provides a detailed look into the security design issues involved in the design and implementation of the TRW A1 Secure Database Management System. The system architecture is presented as well as design issues relevant to deciding whether DBMS functions are to be included within the trusted computing base (TCB).

1 Introduction

TRW's A1 Secure Database Management System is a multilevel secure relational database management system (DBMS) that is currently being developed under the Advanced Secure DBMS (ASD) IR&D project by the Defense Systems Group of TRW. This paper will describe the security architecture of the A1 Secure DBMS and discuss the major security concerns

The objective of the ASD project is to ultimately achieve a high performance secure DBMS that will be successfully evaluated as meeting the A1 standards as defined by the yet to be released Trusted DBMS Interpretation (TDI) of DoD 5200.28-STD (the Orange book).

The A1 Secure DBMS project is reusing much of the code from an earlier DBMS project that was unique in that its basic architecture mirrored the classical security kernel and multiply instantiated untrusted DBMS server design. In this paper, this initial, DBMS is called Version 0.

The A1 Secure DBMS is being written to the greatest extent possible in Ada¹. This is consistent with the fact that Version 0 was written in Ada.

The A1 Secure DBMS is not currently a product.

¹Ada is a registered trademark of the U.S. Government, Ada Joint Program Office

It is a prototype development that is attempting to do research and advanced development on secure DBMS technology appropriate to the A1 level of evaluation by the National Computer Security Center (NCSC). While initial development of the A1 Secure DBMS is taking place prior to the availability of the Trusted Database Interpretation, it is anticipated that the DBMS A1 criteria would strongly follow the existing operating system A1 criteria.

The A1 Secure DBMS will ultimately be hosted on an A1 secure operating system under development by TRW. However, since Version 0 runs on UNIX² on the Sun Workstation, the initial development of the A1 Secure DBMS is also under UNIX on the Sun Workstations. When the A1 secure operating system becomes available, the A1 Secure DBMS will be ported to it.

The remainder of this paper will consider three aspects of the A1 Secure DBMS design.

Section 2 will present the operating modes in which the A1 Secure DBMS can be operated. Section 3 will present each of the security and integrity requirements guiding the A1 Secure DBMS design. Section 4 will discuss the architecture of the A1 Secure DBMS.

2 The A1 Secure DBMS Modes of Operation

The A1 Secure DBMS can be operated in three different modes illustrated by figures 1, 2 and 3.

Under the first mode of operations, the A1 Secure DBMS functions as a DBMS server on a local area network. Under the second mode of operations, the A1 Secure DBMS can serve as a backend DBMS for various single level or multilevel host computers. Under the final mode of operations, the A1 Secure

²Trade Mark of AT&T.

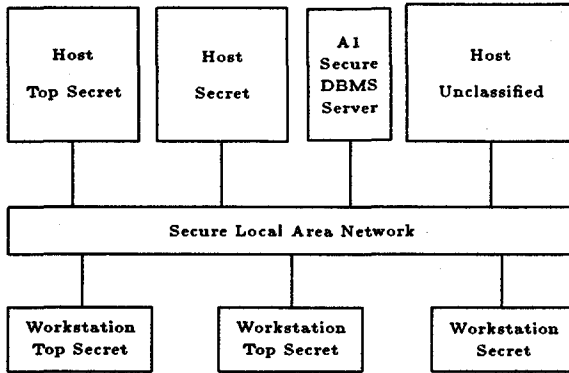


Figure 1: Local Area Network Mode

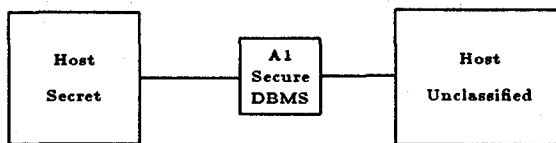


Figure 2: Back-end Mode

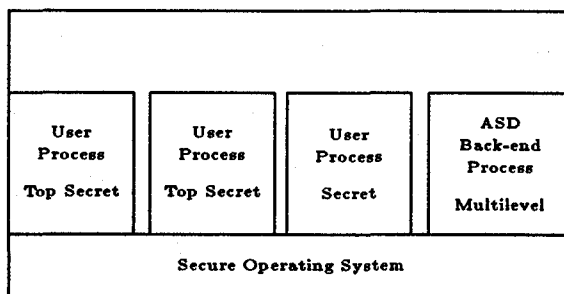


Figure 3: Stand-Alone Mode

DBMS can serve as a host resident DBMS within a multilevel host running an A1 secure operating system.

3 The A1 Secure DBMS Security Requirements

This section briefly describes the security policy that the A1 Secure DBMS is designed to enforce, and then presents the basic system security requirements that the system is designed to satisfy.

3.1 The A1 Secure DBMS Security Policy Overview

The mandatory "object" of protection in the A1 Secure DBMS is the tuple of a table³. The mandatory security policy enforced satisfies the Bell and LaPadula security policy model[BELL76].

Tuples inherit their discretionary access from the tables in which they are located. Discretionary access is specified for tables in terms of permissions for access and denials of access. Permissions and denials may be specified with respect to users, groups, or public. The permissions are select, insert, delete and update. Under the current design, the most specific discretionary access specification takes precedence over a less specific specification and a denial (at a given specificity) takes precedence over a permission. A user is more specific than a group which is more specific than public.

The A1 Secure DBMS also enforces the Biba [BIBA77] integrity model which states that a subject may read a tuple if and only if the integrity level of the tuple dominates the integrity level of the subject. A subject may write a tuple if and only if the integrity level of the subject dominates the integrity level of the tuple.

3.2 ASD Users

The A1 Secure DBMS recognizes four different types of user privilege: normal user, database administrator (DBA), system security officer (SSO), and table

³See [WILS88] and [GAWU88] for an approach in which the view is the object of protection. While the work reported on in these two papers was funded under the same research project as the A1 Secure DBMS, it was not targeted for inclusion in the A1 Secure DBMS.

	Normal User	DBA	SSO	Table Owner
Read, Add, Remove, Modify Row Information	X			
Create Database		X		
Create Tables In Database		X		Future Plans
Add Users To System			X	
Delegate Table Creation Privilege To Owner		Future Plans		
Change Discretionary Access Rights		Interim		Future Plans
Declassification			X	

Figure 4: The A1 Secure DBMS User Privileges

owner (a normal user that created a table). Figure 4 illustrates the current capabilities of the system. "X" indicates current capabilities that are to remain, "interim" indicates current capabilities that are to be changed, and "future plans" indicate ultimate system capabilities.

3.3 The A1 Secure DBMS Security Requirements

This section lists the security requirements that are to be satisfied by the A1 Secure DBMS. For later reference, each requirement is numbered with "R-X", where the X is 1,2, etc. For many of the requirements, subrequirements are identified and indicated by R-X.1, R-X.2, etc. These requirements are based on the requirements of DoD 5200.28-STD, our own concerns relevant to DBMS security, and the particular design used in the A1 Secure DBMS.

R-1 Identification/Authentication. The A1 Secure DBMS shall ensure that all access to data is traceable to a uniquely identified and authenticated user.

R-2 Mandatory Disclosure. The A1 Secure DBMS system shall not disclose data in violation of the A1 Secure DBMS mandatory security and in-

tegrity policy. This includes the following aspects of disclosure prevention:

R-2.1D (disclosure) Mandatory Read Check. Enforcing this satisfies the simple security condition of the Bell and LaPadula model for disclosure.

R-2.1I (Integrity) Mandatory Integrity Check for Read Enforcing this satisfies the Biba model for integrity.

R-2.2D Mandatory Write Down Prevention. Mandatory checks shall be made to ensure that any data stored in the database flows from a process whose security level is equal to that of the data object into which the data is to be stored. Likewise all deletion of data should be performed only by processes whose security level is equal to that of the data object to be deleted. This is a more stringent version of the Bell and LaPadula *-property enforcement, since it does not permit the upward writing of data as does that model. The effect of a write up can be accomplished by a read down in a multilevel secure database, hence there was no reason to support a write up/footnote While one might assert this as a universal principle, one could imagine cases in which it would be desirable for the lower level subject to initiate the data transfer and hence a write up (blind or course) might be desirable.

R-2.2I Mandatory Integrity - Writing. This enforces the Biba integrity prohibition against low integrity processes writing into high integrity objects.

R-2.3D Mandatory Intermixing Prevention. Trusted functions that handle data at multiple security levels must ensure that the data does not become intermixed and that if the data is sent to an untrusted subject that the untrusted subject is at the appropriate level for the data.

R-2.3I Mandatory Intermixing Prevention. Trusted functions that handle data at multiple integrity levels must ensure that the data does not become intermixed and that if the data is sent to untrusted code that the untrusted code is at the appropriate level for the data.

R-3 Discretionary Disclosure. The A1 Secure DBMS shall not disclose data in violation of the A1 Secure DBMS discretionary security policy. This requires that the following trusted functions be performed:

R-3.1 Discretionary Operations Check. A discretionary check shall be made to ensure that the user has authorized discretionary access to invoke particular operations, including retrieval, insertion, deletion, and modification.

R-3.2 Discretionary Data Check. A discretionary check shall be made to ensure that the user has authorized discretionary access for the table into which the data is to be stored.

R-3.3 Discretionary Change in Privilege. Only the database administrator in the initial version of the A1 Secure DBMS and the owner of a table in the final version of the A1 Secure DBMS shall be able to change the discretionary access of a table.

R-3.4 Discretionary Intermixing Prevention. The system shall ensure that data with different discretionary privileges, having satisfied R-3.2, that is destined for two different users does not become intermixed such that a user gains access to data to which he is not authorized.

R-3.5 Changes to Discretionary Control. There is no trusted path requirement for discretionary control. According to transaction 54 of the interpretations forum of the National Computer Security Center, "Trusted path is not required for actions on DAC mechanisms." Hence, the use of untrusted software is permissible.

R-4 Always Invoked. The A1 Secure DBMS shall enforce the "Always Invoked" property of the security reference monitor [ANDE72] with respect to DBMS controlled data. This has subrequirements:

R-4.1 Access Checks Always Invoked. Mandatory and discretionary security enforcement shall be applied to every access to the database.

R-4.2 Audit Capture Always Invokable. The audit function shall not be capable of being bypassed.

R-5 Tamper Proof. There shall be no way for untrusted code to corrupt the A1 Secure DBMS reference monitor nor data used by the A1 Secure DBMS reference monitor to make security relevant decisions. This is the tamper proof requirement of the security reference monitor. This includes the concept of a trusted path where required in the A1 Secure DBMS design. It also includes preventing untrusted code from sending authenticated user IDs to trusted code, bypassing the trusted identification/authentication function. This latter prohibition is called spoofing prevention.

R-5.1 Modification Protection of Trusted Code. The trusted DBMS code shall be protected from any modification.

R-5.2 Security/Integrity Relevant Data Protected. Any security or integrity relevant data that is used in making a security or integrity decision shall be protected from unauthorized modification (that could lead to a violation of the mandatory or discretionary DBMS security or integrity policy) by untrusted code.

R-5.3 Security Relevant Commands Protected. Security or integrity relevant commands (such as level changes) and responses that could lead to a violation of the mandatory security policy shall be protected from modification. This is the trusted path property applied to commands.

R-5.4 Alias Prevention. The system shall ensure that aliases for stored data are not created such that the same protected data is viewed by the system as two distinct tables with two distinct and possibly different sets of discretionary protection specifications.

R-6 Covert Channels. The A1 Secure DBMS system shall minimize the bandwidth of any covert storage or timing channels. The following are the design parameters for the A1 Secure DBMS.

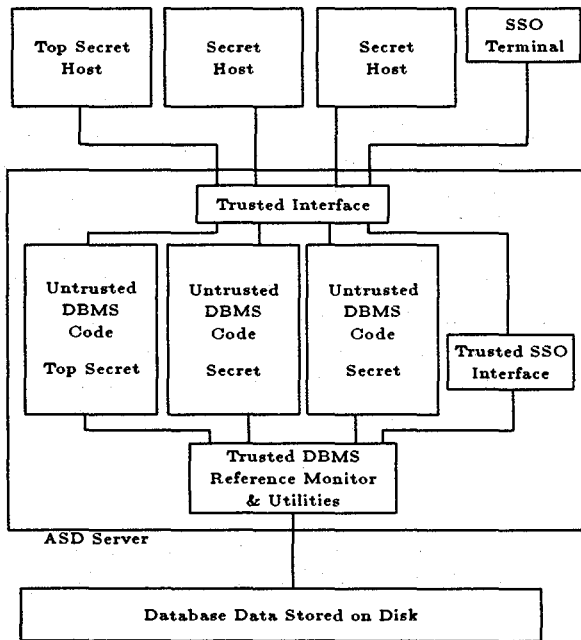


Figure 5: Top Level A1 Secure DBMS Architecture

R-6.1 Absolute Size. The interface of untrusted functions to trusted functions shall not provide any covert channels which exceed 1 bit/second.

R-6.2 Audit Requirements. Those covert channels that exceed 0.1 bit per second shall be audited.

R-7 Audit. The A1 Secure DBMS system shall be capable of capturing the necessary audit data described in DoD 5200.28-STD[DOD85], including immediate notification of the system security officer when thresholds are exceeded.

R-8 Integrity. The A1 Secure DBMS system shall enforce the A1 Secure DBMS integrity policy.

4 A1 Secure DBMS Architecture

Figure 5 illustrates the A1 Secure DBMS architecture. The lines between the hosts at the top of the figure and the A1 Secure DBMS Server box in the middle represent the means (local area network, direct point-to-point cable, or process call) by which an

application process communicates with the DBMS. A query is formulated in the host (or application process if the A1 Secure DBMS is used in stand-alone mode) and sent to the A1 Secure DBMS server. The trusted interface ensures that the request is serviced by the appropriately classified untrusted DBMS code within the Server. This code processes the request and makes calls on trusted DBMS Reference Monitor code within the Server to actually retrieve the data. Various trusted utilities are present in the system to create and maintain the A1 Secure DBMS database.

Under the A1 Secure DBMS design, multiple instantiations of the untrusted DBMS code run, each at the same level as the host application process that it is supporting. The untrusted DBMS code is considered as an untrusted process. This process is only given that data permitted according to the A1 Secure DBMS security policy. Hence, the untrusted DBMS code plays no role in mandatory security enforcement. It is only given access to data which it dominates in security level. It can only write objects at the same level as the process in which it is currently executing. The security levels of newly created tuples are equal to the security level of the untrusted DBMS process that requested the tuple creation.

The A1 Secure DBMS server can be divided into three parts: trusted DBMS reference monitor code, untrusted DBMS server code, and trusted DBMS utilities. Each of these parts contain the following functions:

1. Trusted DBMS Reference Monitor
 - Network Interface
 - Master
 - Table Manager
 - Operator Manager
 - System Table Manager
 - Transaction Manager
 - Disk Manager
 - Index Manager
 - Lock Manager
 - Buffer Manager
 - System Security Officer Interface
 - Audit
2. Untrusted DBMS Component
 - Sequencer
 - Parser

- Execute
- Decision
- Backend Network Interface

3. Trusted DBMS Utilities

- Create Database
- Consistency Checker
- Dump Database
- Load Database

In the following subsections, the purpose of each function will briefly be described along with an indication of the security requirements that must be satisfied by each function. The security requirements will be described in terms of those security requirements previously presented. Those functions which are trusted are said to be part of the A1 Secure DBMS trusted computing base (TCB) while those functions which have no security requirements are considered to be untrusted and thus not part of the TCB.

4.1 Trusted DBMS Reference Monitor Code

Since the A1 Secure DBMS will operate under the control of a secure operating system, some security functions that are normally associated with a secure system are not part of the A1 Secure DBMS, but are provided by the secure operating system. Identification and authentication is a primary example. In what follows, only the trusted A1 Secure DBMS Reference Monitor functions are described.

Network Interface. A trusted network interface exists to ensure that messages are correctly routed to either to Master or one of the untrusted DBMS server processes. This function will either be totally trusted, or will contain both trusted and untrusted code. Under the latter approach, a trusted low level function determine the security and integrity level (assuming low if none) of the received message. Then, it will send the message to an untrusted network handler classified at the same level of the message. This untrusted network handler will provide the remainder of the protocol processing required by the network. If these functions are handled by a trusted network, then they will not have to be part of the A1 Secure DBMS. This area is still under consideration. The trusted function must address R-1, R-2.3D&I and R-3.4

Master. Master performs the following functions

1. Intercepts all connection requests from the front-end to the A1 Secure DBMS in the back-end. It must be trusted to ensure that messages do not become intermixed in violation of R-2.3D&I and R3.4, since the messages themselves may contain sensitive data embedded in the connection request by untrusted code within the untrusted front-end processes.
2. Upon receipt of a database open command from the front-end, Master performs the following actions to establish an appropriately classified untrusted DBMS process in the back-end:
 - Performs required identification and authentication actions to establish the identity and security level of the front-end user or process requesting service. This must satisfy R-1.
 - Has the underlying secure operating system create an untrusted DBMS process, at the level of the requesting user process, to service requests from the front-end process that requested the opening of the database. This is in support of R-2.1D&I, and R-2.2D&I.
 - If correctly authenticated, establishes a communications link, at the level of the requesting front-end process, between the untrusted DBMS process and the client process on the front-end that requested the open. This is in support of R-2.1D&I.

Initial identification and authentication of users is assumed to be handled by the secure operating system on which the A1 Secure DBMS operates. But the DBMS also checks to be sure the user is authorized to access the database.

Table Manager. The primary function of Table Manager is to initialize and terminate access to those tables maintained by the A1 Secure DBMS. The functions, including the security relevant functions, that it performs are as follows:

1. Ensure that access to a table satisfies the discretionary access requirements. This satisfies R-3.1 and R-3.2.
2. Ensure that only the owner of a table can change the table's access control list. This satisfies R-3.3.

3. Ensure that the following events can be audited (R-7):

- failed access attempt due to no discretionary access
- attempt to change access control list by other than the owner

4. Creates and then maintains a description for all currently open tables.

5. Acquires appropriate shared read lock or exclusive write lock on table.

Operator Manager. The Operator Manager function provides tuple access: retrieve, add, modify or replace. It can access a tuple based on a conjunction of search arguments or a tuple identifier.

The primary functions, security relevant and otherwise, that it performs are as follows:

1. Search for or modify the next tuple that satisfies specified search arguments for the specified table.
2. Ensure that each tuple to be returned satisfies the search arguments as well as the mandatory security policy, such that the level of the tuple to be returned is at a level (security and integrity) appropriate to the user process making the request. This satisfies R-2.1D&I.
3. Ensure that for all modified tuples or added or deleted tuples, the level of the tuple satisfied both the integrity and security policy with respect to the user process making the request. This satisfies R-2.2D&I.
4. Change the security level of a tuple (under control of the System Security Officer).

System Table Manager. The System Table Manager function provides rapid access to the information maintained in system support tables. These are the tables that describe the database.

The A1 Secure DBMS contains the following system tables:

SYS_OBJECTS. Stores information about each table in the database.

SYS_COLUMNS. Stores information about each attribute in the database.

SYS_USER. Stores information about the users of the DBMS.

SYS_PROTECTIONS. Stores information about the discretionary protection provided for each table.

SYS_ERROR. Stores the error message text.

SYS_HOSTS. Stores the name and network address of each host that is using the DBMS.

The security relevant aspects of the System Table Manager function will be analyzed in terms of the tables it manages. In theory, a table can be security relevant for one of the following reasons:

1. Table information is classified, and must be protected from viewing by processes whose security level does not dominate the level of the data. Since System Table Manager calls Operator Manager to actually retrieve the tuple of the table, the necessary mandatory security and integrity check (R-2.1D&I) is handled by Operator Manager and does not have to be performed by System Table Manager.
2. In theory if the DBMS is not carefully designed, modification to some of the information in the table could lead to information being disclosed in violation of the A1 Secure DBMS security policy or integrity being violated. For this reason, R-5.2 must be satisfied. In the current implementation, security and integrity relevant information such as security level indicators are not stored as part of the table, hence System Table Manager has no access to any information that could lead to a mandatory violation. It does have access to information that could lead to a discretionary violation.
3. In theory, if the DBMS is not carefully designed, some of the information in the table could be changed by actions at one security level and such changes could be observed by untrusted processes at a lower security level. This could lead to a covert channel. R-2.2, R-6.1 and R-6.2 must be satisfied. Since System Table Manager uses Operator Manager to perform these writes, if System Table Manager were multiply instantiated by security level, Operator Manager could ensure that data was viewable only in a tuple labeled at the level of the version of System Table Manager performing the access. Otherwise, System Table Manager must be trusted. This issue is still under consideration.

Transaction Manager. The Transaction Manager supports DBMS transactions. To do this, it must record all database update operations in a system log.

There are two approaches for the design of the Transaction Manager. The first approach it have an instantiation of the Transaction Manager run as part of the untrusted DBMS server processes. Under this approach, all of the transaction data is stored at the level of the particular DBMS server process being supported, and the transaction can be untrusted.

Under the second approach, used in the A1 Secure DBMS, there is a single trusted Transaction Manager. While this approach does have more trusted code than under the first approach, it centralizes the storage of the transaction records and avoids the proliferation of transaction records, at potentially many different security levels, that would exist under the first option. Under this approach, the Transaction Manager would have to be trusted since it records and retrieves data that is potentially at multiple security levels. Hence, it is trusted not to: intermix data in violation of mandatory policy (R-2.3D&I) or change the security or integrity level of data (R-5.2).

Disk Manager. The Disk Manager function manages the allocation of disk space. It does not have access to the disk. It accesses the disk via buffer Manager, and then, has access only to tables that indicate the status of disk allocation, hence it is not security relevant because of disclosure concerns.

The other possible security concern is that the Disk Manager, if untrusted, could use the disk allocation tables as a storage channel. This requires that untrusted code with access to sensitive data have access to the Disk Manager. But, the Disk Manager provides service only to the Operator Manager, a trusted function. It can not be accessed directly by untrusted code, thus at best, it could only be used by untrusted code using the Operator Manager to fill all storage and then release storage, thus causing lower level processes to block, waiting for additional storage. This represent a "1", "0" channel that is estimated to be of very low bandwidth. The bandwidth could be reduced by placing delays in Operator Manager to limit the responsiveness of the system to messages indicating that no storage is available and/or by auditing these situations.

From the above, it would appear that the Disk Manager need not be in the TCB since the trusted Operator Manager can limit whatever leakage can be

causes through allocation and deallocation. Also, as noted, since the Disk Manager need not have any access to sensitive data it would not even be the source of this leakage.

One argument for making Disk Manager part of the TCB is performance. If Disk Manager were untrusted, then there would be a process switch each time it were called by Operator Manager. The counter argument to this is that it is only called when new disk space needs to be allocated or old space deallocated. In general, this should not occur too often. This is a design issue that is still under consideration.

Index Manager The A1 Secure DBMS is not yet a complete DBMS implementation and its Index Manager has not been implemented.

The Index Manager handles indexes, including their creation, deletion and use. Index Manager is in the TCB since the indexes may themselves contain multilevel data. The Index Manager will enforce R-2.1D&I and R-2.3D&I.

Consideration was given to storing all of the indexes at system high under the control of an untrusted index Manager. This approach is unacceptable, since it would lead to extremely high covert channels. Such channels arise since the index data would have to be downgraded to the requesting process, and such information, since it would be under the control of untrusted code could be used as a high bandwidth leakage channel.

Consideration was also given to multi-instantiating Index Manager. While this would lead to an untrusted Index Manager, it would also lead to the need for a separate index for each level, leading to very slow response for queries that span multiple levels.

Lock Manager The Lock Manager function performs lock management including granting, releasing, and enforcing locks. Locks are placed on the entire table under the current design. Future enhancements will add page locks.

Under the current implementation, Lock Manager is called by Table Manager, a trusted function, to set the locks, and by Transaction Manager, a trusted function, to release the locks at the completion of a transaction.

Lock Manager does not require access to sensitive data, nor does it require access to security relevant data whose modification could ultimately lead

to unauthorized disclosure in violation of the DBMS security policy. This leaves covert channels or performance as the only justification for including Lock Manager in the TCB. Each of these will be considered.

Since Lock Manager does not have any direct interface with untrusted code, any covert channels provided by Lock Manager must be via indirect channels through other trusted code. The primary, and we believe only, example of this is the classical channel provided by a process at a high security level locking a table (to prevent the data from being written while the high process is reading it) and a process at a lower level sensing that the table has been locked.

There are at least two ways that this covert channel could be controlled. The first is to place a governor on the rate at which locks can be changed. The governor can be adjusted to ensure that any covert channel is less than any desired rate, say 1 bit per second. This, of course, has a performance impact since it wastes some time between when the lock is given up and the governor permits the effect of the lock release to become effective. However, since the governor would only have to be applied to write locks (the only locks that can be sensed at a lower security level) this would not cause a major performance problem for those systems where the ratio to reads to writes is high. If required, this channel could be audited, although one would have to be careful not to inundate the audit system with audit data on every write lock. Ideally, one would only record those write locks that caused the delay of writing by a process at a lower security level.

Under the current A1 Secure DBMS design, locks are placed on entire tables, so a single query would involve only a few tables and thus a few locks. However, future plans call for setting locks at the page level. Now, a single query might involve the setting and release of a large number of locks. Under this locking granularity, adding additional delay to each lock could be highly detrimental. One could consider adding the delay to the query, not just the lock. While this counters the vulnerability to Trojan horse initiated covert channels between hosts, it does not counter the vulnerability to Trojan horse initiated covert channels between untrusted DBMS server processes, since they could choose to set locks at a granularity below the query level. This aspect of the lock design is still under investigation.

The second approach to counter the lock based covert channel is to use event counters. Under this approach, reported in [HINK75], a counter would

be incremented (using modulo arithmetic to handle wrap-around) and a write-in-progress flag set each time a modification to a table commenced. The counter and write flag would be checked at the beginning and end of each read to the table. If the write flag was set, no read would occur. If the write flag was not set, the read would be performed. If the counter changed during the read, then the read would be treated as a failed transaction and redone. In this way, the covert channel is closed since at most high security level processes are delayed by low processes, but low processes would not be delayed by high processes.

Under the first approach, Lock Manager would have trusted code to control the covert channel rate and possibly provide audit information as required. Under the second approach, it would not. However, now, low security level activity could deny service to high security level activity. This is a trade-off that will be addressed when lower granularity locks are added.

Buffer Manager. The Buffer Manager function manages the buffers that are shared among the server processes. Since it handles the actual pages which may contain data at multiple security levels, Buffer Manager must be trusted. Access to the pages controlled by Buffer Manager is through Operator Manager, hence Buffer Manager is not directly responsible for enforcing the A1 Secure DBMS security policy. However Buffer Manager must ensure that data under its control does not become intermixed (R-2.3D&I) and that the security and integrity labels on tuples are not modified (R-5.2).

System Security Officer Interface. The System Security Officer Interface provides trusted support within the A1 Secure DBMS back-end for those operations required by the System Security Officer. This interface is analogous to the untrusted DBMS code component, except that this interface contains only trusted code and supports the System Security Officer functions. The general functions supported by this interface include the following: user administration, audit, object reclassification, and aggregation detection (not trusted in the current design).

The aggregation detection function will permit the SSO to establish queries that can be used to detect the existence of conditions that have been identified as leading to an aggregation problem. Such an aggregation problem exists if lower security level data can be used to infer higher level data. See [HINK88] for

additional details about how the aggregation problem is addressed.

It should be noted that this function does not include some things normally associated with System Security Officers since these are assumed to be performed by the A1 secure operating system's System Security Officer function, rather than the A1 Secure DBMS's System Security Officer function.

From a mandatory security perspective, the only SSO function that is security relevant is object reclassification, and this must satisfy the trusted path requirement of R-5.3 and the requirement that no unauthorized modification be made to security relevant data in satisfaction of R-5.2. Changes to the integrity labels will be handled in an analogous way.

Audit. The audit function is not currently implemented. In the design, the audit function is called from various programs within the TCB to collect security relevant audit data. It stores this audit information in a system high relation accessible only to the System Security Officer. Protection of audit data from system high users can be via the discretionary access controls (SSO only).

Audit is security relevant only to the extent that failures in the audit function could cover up activity that is attempting to compromise the system. Under the requirements of DoD 5200.28-STD, the system security officer must be notified when thresholds are exceeded. While in one sense the audit function can be viewed as a second order function in that its incorrect operation will not in itself lead to disclosure, the requirement for real time notification does place audit close to the front lines of security protection. Because of this second order issue, a question was raised at the 1986 National Computer Security Center Invitational Workshop on Database Security whether audit had to satisfy the same stringent requirements as the other first order trusted components of the system (i.e., in the case of A1 systems must the audit code have to be formally specified and verified). This is an open question that is not a DBMS specific issue. Its solution for DBMSs must await the Trusted Database Interpretation.

Another reason, which we believe to be incorrect, for trusting the audit system is it may contain multi-level information. This is an incorrect argument for making the audit function trusted since audit data will be stored as part of a system high table, accessible only by the system security officer. Since the audit table is system high, there is no require-

ment to maintain labels. Likewise, there is no disclosure threat since there is no outlet for the sensitive data other than to the System Security Office who is trusted.

From the above discussion, Audit must satisfy the following security requirements: R-5.2 (protect the audit data from modification), R-4.2 (ensure that audit function can not be bypassed), and capture necessary security relevant actions in support of R-7.

One additional design issue exists relative to audit. As mentioned above, the criteria interpretations forum of the National Computer Security Center has extended the DoD 5200.28-STD (for B3 and A1) to require that Audit take real time action. The A1 Secure DBMS approach to real time notification will be addressed in the future.

4.2 Untrusted DBMS Code Component

This section describes the functions within the untrusted DBMS code of the A1 Secure DBMS. This section will describe their functional properties and the justification for not including them as part of the TCB.

Sequencer. The Sequencer coordinates the processing of the DBMS request. The only security relevant operation whose incorrect operation could potentially lead to disclosure, and then only with respect to discretionary security policy, is changes to the discretionary access control list. However, since these actions do not require a trusted path, there is no DoD 5200.28-STD required trusted software within the sequencer. While the sequencer needs to support R-3.5, it need not perform this support with trusted software. Hence, Sequencer need not be part of the TCB.

Parser. The Parser takes DBMS commands in the query language and translates them into in internal form, a sequence tree.

The Parser must be modified to accept commands which change the discretionary access in support of R-3.5. However, based on the same arguments used for the sequencer, the parser need not be part of the TCB.

Execute. The Execute function executes the command tree formed by the Parser.

For the same arguments provided in Sequencer, the Execute function need not be trusted.

Decision. The Decision function develops a strategy to be used for execution of each command tree. It attempts to minimize the resources needed to execute the query tree. In the final system, decision will contain database statistics which may need to be kept by level or all moved to the TCB.

For the same arguments provided previously, the Decision function need not be trusted.

Backend Network Interface. The Backend Network Interface (BENI) function sends and receives all data arriving at or leaving each of the untrusted DBMS processes. BENI is untrusted, since one instantiation of BENI exists for each untrusted server process.

4.3 Trusted DBMS Utilities

This section will consider the functional and security properties of the trusted DBMS utilities.

Create Database. This function formats the logical disk and initializes the system tables, creating a legal, empty database.

This function creates the SYS.OBJECTS and SYS.COLUMNS tables. It then fills in the SYS.OBJECTS and SYS.COLUMNS information for the system tables. It then reads in information to fill in entries for SYS.USER, SYS.HOST, and SYS.ERROR tables.

The Create Database function does not have any access to sensitive data since it creates an empty database. It is assumed that the secure operating system on which the A1 Secure runs ensures that the Create Database function has no access to sensitive data that exists under the control of the operating system.

One justification for making Create Database trusted is that it reads in data to fill the SYS.USER and SYS.HOST tables, and this data could be sensitive, and hence must be protected from disclosure. However, this table filling step could be done in a

second step, by a different process, thus reducing the amount of trusted code required.

The second justification for considering Create Database trusted is that it can initialize the A1 Secure DBMS structure in such a way that it could lead other DBMS trusted code which does have access to sensitive data to violate the DBMS security policy.

There are three different ways that Create Database could, in theory, confuse other A1 Secure DBMS trusted code: create extra structures (e.g. additional tables or attributes over and above those required), create less structure than required (e.g. leave out a table or an attribute), create bogus data in a table.

Each of these will be considered in turn.

Additional Structure. If additional system tables are added, they will be ignored since none of the trusted DBMS reference monitor code which uses the system tables will know anything about them. The additional tables will not be considered user defined tables since, under this "Additional Structure" attack, they are not included in any system tables (that attack will be included under the "Bogus Data" attack analysis). Thus, the conclusion is that if the Create Database function malfunctioned and added additional tables beyond those required by the A1 Secure DBMS, this would not lead to a unauthorized disclosure⁴ since the extraneous tables would be ignored.

Less Structure. If Create Database left out some required tables, then this would cause a run time error since expected data structures would be missing. Analysis will have to be applied to the code to ensure that no disclosure can occur and that the system halts on such an error.

Bogus Data. Under this attack, we assume that bogus data could be added to any of the system tables. If bogus data is added to the SYS.OBJECTS table, then it would define non-existent tables. The main concern here is that if later, a real table is defined with the same name, then there is the possibility that a user accessing the real table might inherit the access privileges associated with the bogus table of the same name. This would not lead to a mandatory violation since each tuple of whatever table is

⁴Unless the A1 Secure DBMS code could also be modified, which it can not be online.

accessed bears a security level label which is checked on each access. However, this could lead to a discretionary violation since the access privileges for a particular user for the bogus table might be more liberal than those for that user for the newly created table. This is an example of the general vulnerability if duplicate table names are permitted. While the Create Database function could prevent this, it need not be done here. The reasons for this is that the Create Table function already ensures that each new table is unique. Hence, there is no possibility of a user created table having an alias description.

If bogus data is added to the SYS_COLUMNS table, this has the effect of defining additional system attributes. One vulnerability could occur if an entry in this table refers to a security level attribute of a table not yet created, but defined to be at a position different from the normal position of such an attribute. Since attributes are associated with their objects through object identifiers, eventually a table will be defined to which this attribute description refers. Upon access, it is possible that the bogus security level attribute definition will be used rather than the real one, leading to a misclassification of the tuples in the table. Under the current A1 Secure DBMS design, this vulnerability is countered by not having the security level for a tuple be in an attribute described in SYS_COLUMN. The security level label is considered to be control data associated with the tuple that precedes the attribute data for the tuple. Secondly, the creation of tables within the A1 Secure DBMS must be designed to handle cases of column name conflicts.

Create Database Conclusions. From the above analysis, the concern is not with any mandatory vulnerability, but with run time errors (in the less structure attack) and discretionary violations (in the bogus data attack). Because of these concerns, it is desirable that Create Database be restricted to creating only those entries in the SYS_OBJECT and SYS_PROTECTIONS tables that are desired, and no more. In all cases, the only property that must be enforced by the Create Database function is R-5.4. R-5.2 (no modification of data used to make access decisions) is enforced by the general design of the A1 Secure DBMS which places the security level label outside of the bounds of the table.

Consistency Checker. This function is not currently implemented in the A1 Secure DBMS.

This function checks the internal physical integrity

of the database. It is intended that it check the physical structure of the database, not the integrity of the data stored in the database. If this function has access to the actual disk storage, it must be trusted not to modify the security relevant data stored on disk, including the security level labels of the data stored on disk, the security relevant entries in the SYS_PROTECTIONS table, and information used for discretionary security enforcement including the object identifiers stored on each page that prevent the alias attack previously described. It thus must preserve R-5.2 and R-5.4. It must also preserve R-2.3D&I and R-3.4 (Intermixing protection). However, under the control of a secure operating this function can be made read-only, and thus untrusted as long as it can not communicate any of the data that it reads to untrusted processes, and this can be guaranteed by the operating system by running it system high.

Dump Database. This function dumps the entire database to a specified operating system file. Since this function has access to the actual data stored on disk, it must be trusted not to modify the security or integrity relevant data as it is dumped to the file. It must preserve R-5.2 and R-5.4. It must also preserve R-2.3D&I and R-3.4 (Intermixing protection).

Load Database. This function reloads the database from a previous dump. It must be trusted for the same reasons as Dump Database.

5 Conclusions

We have presented an in depth description of the A1 Secure DBMS design, our requirements and an analysis of the trusted versus untrusted system components. The A1 Secure DBMS design can be implemented within 2.5 to 3 years with current technology. A class A1 multilevel secure relational DBMS based on our design would meet the needs of a variety of multilevel applications and end users.

References

- [ANDE72] Anderson, J.P., "Computer Security Technology Planning Study," ESD-TR-73-51, vol. I, AD-758 206, ESD/AFSC, Hanscom AFB, Bedford, Mass., 1972.

- [BELL76] Bell, D.E. and LaPadula, "Secure Computer Systems: Unified Exposition and Multilevel Interpretation, Report ESD-TR-75-306, MITRE Corporation, Bedford, Mass., March 1976.
- [BIBA77] Biba, K. J., "Integrity Considerations for Secure Computer Systems", MTR-3153, MITRE Corporation, Bedford, Mass., April 1977.
- [DENN87] Denning, D.E. and R. Schell, "Secure Distributed Data Views (SeaView) - The SeaView Formal Security Policy Model," SRI International, July, 1987.
- [DOD85] "Department of Defense Trusted Computer System Evaluation Criteria," December 1985.
- [GAWU88] Garvey, Crista and Amy Wu, "ASD.Views," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, 1988.
- [HINK75] Hinke, Thomas H. and Marvin Schaefer, "Secure Data Management System," RADC-TR-266, Rome Air Development Center, Griffiss AFB, N.Y., Nov. 1975 (NTIS AD A019201).
- [HINK88] Hinke, Thomas H., "Inference Aggregation Detection In Database Management Systems," Proceedings, IEEE Conference on Security and Privacy, Oakland, 1988.
- [JENS87] Jensen, Nancy, "A1 Secure DBMS Prototype - System Security Officer (SSO) Functional Design," TRW ASD Project Report, 1987.
- [KUG] ASD Kernel Users Guide.
- [WILS88] Wilson, Jackson, "Views as the Security Objects in a Multilevel Secure Relational Database Management System," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, 1988.
- [WILS87] Wilson, Jackson, "ASD Security Policy", TRW internal report, July 1987

Evolution of A Model for Computer Integrity

(Session A: Modeling Integrity)

by

David D. Clark
Senior Research Scientist
MIT Laboratory for Computer Science

and

David R. Wilson, National Director
Information Systems Consulting Services
Ernst & Whinney

11th National Computer Security Conference
October 17-20, 1988
Baltimore, Maryland

This is a working draft of a paper for presentation at the National Computer Security Conference and is not to be reproduced or used for any other purposes.

© 1988 Ernst & Whinney

Evolution of a Model for Computer Integrity

I. Background

More than 18 months ago we presented a model for data integrity in our paper, "A Comparison of Commercial and Military Computer Security Policies," presented at the annual IEEE Symposium on Security and Privacy [Clark and Wilson]. That model, since known as "Clark-Wilson," encouraged the information systems and computer security communities to press forward with integrity-related research. We now wish to give some sense of how the research is going, and, in light of that research, to clarify certain issues raised in our original paper. Those issues involve defining a context for integrity and defining the concept as an aspect of computer security, achieving "real-world" integrity, identifying the features of systems in which integrity is the main security goal, and expanding the U.S. Department of Defense "Orange Book" [DoD] disclosure model to embrace the idea of integrity.

The original paper has generated some follow-on activities. A Workshop on Integrity Policy in Computer Information Systems (WIPCIS) was convened October 27-29, 1987, at Bently College [WIPCIS]. It was attended by more than fifty researchers and security professionals. A draft of the workshop report was published and distributed at the 1988 IEEE Symposium on Security and Privacy. Three papers were presented at the 1988 symposium describing potential implementations of the Clark-Wilson integrity model [Karger; Lee; Wiseman, et al.]. An informal session on the future of the model also was held.

The National Bureau of Standards has sustained an interest in the Clark-Wilson Model by releasing the official WIPCIS report and by making integrity security one of its priorities for the Institute for Computer Science and Technology (ICST). The NBS has established the Computer and Telecommunications Security (CTS) Council to identify and study key issues and common requirements in the CTS area; a Working Group has been established within the Council to study the area of data integrity. Working Group leader Bob Courtney has recently summarized the results of the group's study. NBS also will hold a follow-on integrity workshop, January 25-27, 1989.

Other integrity model-related activity includes the recent Canadian Trusted Computer Product Evaluation Criteria Workshop, held in Ottawa, Canada [Canadian], at which the issues raised by Clark-Wilson were discussed in relation to the U.S. Department of Defense "Orange Book."

II. Context and Definition of Integrity

Because of the precedent set by the United States Department of Defense Trusted Computer System Evaluation Criteria, or "Orange Book" [DoD], many of the implementation schemes for the Clark-Wilson model have focused on computer systems design. This focus has been most necessary and valuable. However, we had intended the Clark-Wilson model as a broader mapping of the issues of integrity that bind real-world concerns to computer system design.

We defined integrity in the original paper as those qualities which give data and systems both internal consistency and a good correspondence to real-world expectations for the systems and data [Clark and Wilson]. Primarily, the expectation of integrity means that systems and data remain predictably constant and change only in highly controlled and structured ways. This concept of integrity is tied to both an internal and an external consistency standard, and is a key element of the Clark-Wilson approach. However, with much work on the subject to date focused more on internal issues, such key concepts as the role of the IVP (Integrity Verification Procedure) and separation of duty have become blurred.

This issue is particularly important because many of the enforcement mechanisms for external consistency require significant internal systems features as part of the basic software and hardware design. For instance, a principle of systems design for separation of duty is that the system must be able to reflect the separation of duty being implemented by application users in real-world environments. This ability to reflect the implemented separation of duty within a system is a complex process which can be greatly simplified if the necessary capabilities are built into the operating environment from the start.

III. Achieving "Real-World" Integrity

By "real-world" we mean the facts, data, and processes outside the computer system which the computer system is expected to reflect, understand, or emulate in some way. Although both internal and external consistency are important, the final test of integrity must be to ensure that the data in the computer is consistent with the world it is intended to represent. If an internal inventory record does not correctly reflect the number of items in stock, it makes little difference if the value of the recorded inventory has been reflected correctly in the company balance sheet.

It stands to reason, then, that integrity controls can never be a matter strictly internal to the computer. A cross-check with the external reality is a central part of integrity control. The computer system can be expected only to preserve the integrity once it has been externally verified.

Methods for Internal Consistency

In our original paper, we described a set of methods for assuring the internal consistency of stored data. This section broadens some of the concepts of internal consistency we introduced in that paper.

Prevention of Change--The simplest method for ensuring the internal consistency of data is to prevent data modification. With this form of control, one need only ensure that the data was correct at one time; since it cannot change (of its own accord) it is possible to trust the data from that time forward.

This mode of control is often the one needed within a network. While data is in transit, it is often sufficient to ensure that it does not change at all. Some form of data check function is often used to verify that data has been delivered intact. This form of control becomes less obvious if the network is expected to perform some sort of format conversion of the data, which suggests that reformatting internal to a network is not consistent with this simple form of consistency control.

Attribution of Change--Another important form of control is to bind the data to its author in an unforgable way. We call this attribution of change, a control which applies to the many sorts of data which do not have a strong internal structure. While accounting records are highly structured internally, an essay on market opportunities is not. With such data, the primary assurance of integrity is the knowledge of authorship, and the assurance that the data has not been modified without the author's knowledge. In this circumstance, a complete log of the data's modification history must be associated with the data, along with the identity of the authors. The system must assure that the data content is exactly that which was provided by the attributed author.

Constraint of Change--For highly structured data such as accounting records, the form of control we call constrained change is applicable. In this mode, the data is modified only by certain programs that have been certified to change the data in constrained ways. We call these programs Transformation Procedures, or TPs [See also Clark and Wilson]. In the example of accounting records, the constraint of double-entry bookkeeping might be enforced: if one account is credited, another must be debited to match.

Partition of Change--The final form of control is partition of change. In this control, the system must ensure that a change is performed by two different people authenticated through user-identifications. Here the system enforces the process whereby no one person has the ability or authority to modify the data and individuals are expected to check each others' work in some manner. The system thus reflects a common business practice, which we describe in our original paper as separation of duty.

In each of the cases given above, TPs are used to maintain a concept of control which enforces internal consistency of the data within the system. These approaches are necessary but not complete. As described

earlier, integrity also requires a correspondence to the outside world. We now discuss three key ways in which a system and data are related to the world they are to represent.

The Integrity Verification Procedure

In our original paper, we introduced the idea of the Integrity Verification Procedure, or IVP. The IVP has a formal relationship to the rest of the model. The proposed proof methodology to demonstrate consistency after running a number of transactions was an inductive one: if each TP takes the system from a valid state to a valid state, then a series of them should take the system through a series of valid states, so the system is finally valid. The necessary condition for this to work is that the system initially be in a valid state. The IVP was proposed to ensure that.

Several people have observed that in a formal sense, this is a redundant feature, as the IVP is just a specific example of a Transformation Procedure, or TP. This observation misses the dual role of the IVP, which is not only to check the internal consistency of the data, but also to verify the consistency between the data and external reality.

Since the IVP checks external as well as internal consistency, it is not just a procedure that is internal to the computer. Instead, it is one of the points where the controls internal to the computer are tied to the larger context of information controls within the organization.

It was observed in one comment that the only reason we need the IVP is that we do not trust the rest of our methodology. Yet this lack of trust does not negate the value of that methodology. Consider again the comparison with a set of accounting records. The books are balanced daily, but once a year, even though good controls have been exercised on normal activities throughout the year, an audit is performed which independently verifies that the records correspond to reality. We need the IVP in the model to capture this idea, accepted in practice, that a system needs a periodic cross-checking.

One other issue associated with the IVP concerns the "reality" a system is measured against. At the WIPCIS Conference there was extensive discussion of integrity domains. When using an IVP to compare a system back to reality, it was recognized that there may be multiple views of that reality depending on the scope of the IVP. These views were defined as integrity domains. It may be necessary, therefore, to label data indicating the particular integrity domain to which it was compared. As systems become large and complex, this comparison with domains will become a necessary process.

The challenge of the IVP is to recognize that those integrity activities that occur outside the computer system must be represented as part of the process of verifying the mechanisms inside a computer whenever possible. There is no way to divorce the outside world from the internal controls on integrity, since integrity is meaningful only in terms of the relation of data to the outside world.

TP Certification

A second major element in assuring the external consistency of data and systems is the TP certification process. This process appears to have two key elements. The first is to assure that the TP does what the specification requests. This includes that source code corresponds to object code, that the TP has been verified and works properly, and that all changes are known and proven as correct.

The second is to assure that the specification for the TP itself corresponds to the "real-world" process it is intended to model. For example, if the TP is to calculate depreciation, the specification should correspond to the correct calculation approach and structure.

When both of these requirements for certification of TPs are taken together, the TP can be assumed to play their part in assuring the integrity of the system and data. These TP certification comments apply both to application TPs and operating system TPs.

Separation of Duty

The separation of duty concepts are the third element confirming that data and systems correspond to the intended real-world model. These concepts have been difficult for everyone to work through and for good reason. Even though they are commonly used in business everyday, they have not been well formalized.

For our purposes, there are several rules concerning separation of duty that are helpful:

- 1) Adequate separation of duty occurs when the custody of elements of a transaction or assets is so subdivided that no one person can commit significant fraud or error without detection or prevention. For instance, to prevent fraudulent transactions, a person who has custody of assets does not also have custody of the accounting record. Similarly, to avoid error, people who keep subsidiary ledgers do not also keep general ledgers.
- 2) The best separation of duty occurs when the people involved in the subdivision of responsibilities have substantially different sets of motives and perspectives. Two people performing critical entry-key verification, or two performing the same act to launch a nuclear missile are examples of the weakest form of duty separation. Stronger forms would include an electronic funds transfer where a clerk in the accounting (records) department initiates a transaction, and a supervisor in the treasury (asset) department releases the transaction.
- 3) Significant breakdowns of the system can occur only when one or more key elements of the separation of duty are violated through the collusion of the people involved. For example, unauthorized access to the computer center is possible through collusion with

the security officer. Without collusion, such access generally is not possible.

- 4) The systems of control cannot be violated by the unilateral actions of one person. This rule is implied by Rule 3.

In the original paper, we acknowledged that the implementation of these rules depends upon the specific way a computer system is implemented in a particular setting. But there is enough generality in the ways in which separation of duty is implemented that it is reasonable to expect the operating systems of a computer to have general enough capabilities to reflect almost any particular implementation of separation of duty.

The next section carries many of these principles forward into a features list for systems and computers.

IV. Computer Support of Data Integrity

So far we have identified a number of mechanisms within the computer that provide for data integrity. In this section we gather these features together, and discuss the manner in which they contribute to the overall integrity goal. It is these features, with others that may be identified in future research, that might be incorporated into some evaluation criteria for future computer systems.

Change Logs and Integrity Labels on Data

To establish that separation of duty has been followed, every important access to a system should be tied to a specific person and logged. This means that the author of data should be recorded in an unforgable way within the data itself, since in many cases the source of the data is the best assurance of the quality of the data. Since data, in general, may undergo a number of modifications as it resides within the system, the record of authorship may need to be a record of the total change history of the object, not just of a single entry. As stated earlier, it may also be necessary to record the integrity status of data by noting the execution of an IVP and the domain used. Some systems currently provide a partial record in the form of two fields, one recording the original author and the other recording the latest author. Application requirements will dictate whether a partial record of this sort is sufficient.

Support of the Access Control Triple

To support the idea introduced above of the constrained change, the system needs to have a mechanism to ensure that data is modified only by selected programs which have been verified in some way to perform only acceptable changes. While TP was intended to capture this idea, the principle of the access control triple is meant to enforce it. The "triple" binds user, program, and data together as a single control object, and thus goes beyond the traditional discretionary control scheme. It may be possible to create the approximate effect of the access control triple by careful

use of traditional access control lists and by representing a program as both subject and object in the permission list, but the result is neither obvious nor precise. For this reason, we believe the access control triple in some form should be a fundamental part of any system oriented towards ensuring the integrity of data.

Mutually Exclusive Transaction Recognition

In support of both the separation of duty concepts and partition of change, some system facility is needed to recognize and label the nature of certain transactions which are inherently mutually exclusive (i.e., from a separation of duty point of view). For instance, in virtually no circumstance would a transaction creation and a transaction authorization be performed by the same individual. In many circumstances transactions which create, delete, or modify information are separated from those that only read information.

To implement this kind recognition capability, there needs to be a rule-set system which labels transactions and then allows for logic to be entered showing which transactions are mutually exclusive from a separation of duty viewpoint. This is analogous to the table-driven user-access controls we see in many systems today but is focused on the TP. With this feature the system could automatically enforce fundamental forms of separation of duty in a manner independent of specific application requirements.

Enhanced User Authentication

Any system concerned with security in any form must have some means to identify the user to the system. The most common method of achieving these goals is the password. However, if the system is concerned with integrity, either through enforcement of partitioned change or attributed change, there are additional requirements for authentication. The system must ensure that the user's identity cannot be forged and that the identity cannot be shared.

This requirement has not received direct attention in most of the literature on computer security, although the concern applies to disclosure controls as well. But the problem is central to integrity control, especially in the area of separation of duty. If for any reason one user gives away his password to another, then that other user can act inside the system as two people, which may permit him to violate the separation of duty rules.

Since violation of the separation of duty rules is the key to corrupting data and committing fraud, any circumvention of authentication must be viewed with great concern. The problem is that the password system as generally implemented does not itself meet the separation of duty rules. The holder of a password can easily and unilaterally invalidate his own identity by making the password easy to guess, by posting it, or by storing it in his own PC. A means is required to prevent the user, through a unilateral action, from circumventing the authentication mechanism.

Passwords should not be considered a realistic authentication method for a system with high expectations for data integrity. Better methods include challenge-response tests involving a device (called the "token") issued to the user that performs a cryptographic transformation on the challenge. Since the transformation is sealed inside the device, it is only possible to loan one's system identity by loaning the actual device, which is a much less trivial action than telling the password. A more rigorous presentation of these principles, called "see-through security," is provided in an article written in 1986 by Andersen, Clark, and Wilson [Andersen].

Control of Privileged Users

To ensure that the basic protections of the system, such as the access control triple, are not violated it is necessary to regulate strictly the actions of privileged users. Privileged users include those who enter access control triples into the system, register new users, or maintain the operating software. The goal of the controls must be that separation of duty is not circumvented. For example, people who can add new users to a system should not know the identifier for those users, and should not be able to change the access rules for those users. Similarly, systems programmers who develop software should not be able to install the software.

Application Program Control

Systems concerned with security must ensure that administrative procedures do not corrupt the system's software. For example, a false release tape can be used to insert changes into an operational system. Similarly, replacement of an object module can cause system behavior that cannot be anticipated by review of the sources.

In a system concerned with data integrity, the control to prevent this sort of corruption must be extended to the application programs as well. Such control, however, constitutes a substantial operational burden, as the bulk of the applications code usually swamps the system itself. For this reason, the system should be provided with standard automated aids to manage application software. These should include tools to enforce source and object synchronization, locks to prevent changes to object code without dual controls, logs of changes, and tools to derive flow diagrams from both source and object code that permit understanding of program flow and changes to that flow.

It should be noted that these sorts of tools are easy to postulate, but require significant effort to define and to put into operation. However, this effort is important not only to integrity, but also to good operating practices in general.

Dynamic Separation of Duty Related to TPs

Separation of duty requires that TPs be divided into sets which are executed by different groups of users. In general, a task (for example,

purchasing something or writing a check) will be designed as a sequence of TPs rather than as one single TP. By requiring the various TPs in this sequence to be performed by different people, separation of duty is achieved.

The simplest way to achieve this separation is to assign different people to different TPs in a static manner, using the access control triple. The system administrator in charge of maintaining the triples is responsible for understanding the way the rules achieve separation of duty and for assigning TPs to individuals. This works, but is limited in functionality because it is often necessary or desirable to reassign people to tasks dynamically.

An alternative approach is for the system to keep track dynamically of the people who have executed the various TPs in the sequence, and ensure, for any particular execution, that proper separation has occurred. This might better model actual, dynamic requirements in the real world.

There are, however, several hard problems to solve in order to implement this function. First, the sequences of valid TPs must be defined in some way. Next, the allowed patterns of separation must be encoded. Finally, there must be some record in the system of each current execution of a sequence, and each TP being executed must be identified as a part of one sequence. WIPCIS participant Bill Murray has described IBM's implementation of such a system [Murray].

V. Evaluation Criteria for Integrity

The previous section outlined a number of features that might be sought in a system oriented toward data integrity. We believe these features should be combined with others used today to provide good support for disclosure control, such as mandatory enforcement of the lattice model. The result would be a unified set of evaluation criteria for systems with respect to integrity and disclosure. In this section, we briefly speculate on an integrated set of evaluation criteria, using the Department of Defense "Orange Book" as a starting point.

The problem is to relate the two sets of features, those for integrity and those for disclosure control. In Figure 1 (see page 10), we have listed the various system features mentioned in the paper and proposed a possible (and very speculative) assignment of these features into a three-division rating system. The lowest listed division, "C", would correspond to the Orange Book "C", and would represent a system with only user-discretionary controls for integrity. The "B" level would require a fairly rigorous set of integrity capabilities within the system which--and as much as possible--are required.

Few additional features are added or made to the "A" level. The major issue at "A" is the degree of certification done to prove the functionality described. Throughout, the Orange Book requirements for operating system certification and other control capabilities are expected to be the same. This table shows only additions.

[Figure 1]

A Mapping of Integrity Features
to Existing Orange Book Requirements

Feature	Division		
	A	B	C
Prevention of Change (e.g., a message authentication code)	R/D	R/D	O/D
Data Labels and Logs			
--Change log	R/M	R/M	R/D
--Change log with attribution	R/M	R/D	O/D
--IVP execution log	R/M	R/M	O/D
--Domain logs for data	O/D	O/D	--
Access Control Triple	R/M	R/M	R/D
Application Program Change Control	R/M	R/M	R/D
Mutually Exclusive Transaction Recognition	R/M	R/M	O/D
Uncircumscribable User Authentication (e.g., tokens)	R/-	R/-	O/-
Controls on Privileged Users	R/M	R/M	O/D
Dynamic Tracking of Separation of Duties	O/D	O/D	--

M = Mandatory

D = Discretionary

O = Optional

R = Required

Figure 1 shows the optional and required features to support levels "C", "B", and "A". It also reflects our views on those controls which are mandatory versus those which are discretionary. In this case we define mandatory as those controls which are unavoidably imposed by the operating system between user and data. Figure 1 depicts five possible combinations describing degrees of control:

- R/M: Control is **required** for this level and its use is **mandatory** across all applications.
- R/D: Control is **required** for this level but its use is **discretionary** (i.e., application-dependent).
- O/D: Control is **optional** at this level and its use is **discretionary** (i.e., application-dependent).
- R/-: Control is **required** at this level but is not directly related to the operating system being imposed between user and data. (Therefore, there is no mandatory or discretionary stipulation at this level.)
- O/-: Control is **optional** at this level and is not directly related to the operating system being imposed between user and data. (Therefore, there is no mandatory or discretionary stipulation at this level.)

The feature that most distinguishes the integrity model of our original paper is the access control triple. Without this feature, the system cannot effectively enforce constrained changes (our transformation procedures, or TPs), which we believe are the key to a broad class of integrity controls. Support of the triple, we argue, is the key indicator of real support for integrity. The triple becomes mandatory at the "B" level.

Several of the other related features would presumably be included at the "B" division in the criteria. For instance, if access control triples are to be enforced, then the change controls on application programs are needed. Similarly, control is needed for the privileged users who create users and TPs if the triple is to be effective. Thus, there is a consistent set of tools that combine to provide a system which relates to integrity in the same way the lattice model of the Orange Book "B" division relates to disclosure control.

The mutually exclusive transaction recognition feature also has been added and allows for the introduction of a mandatory concept of segregation of duty for systems integrity. This concept is introduced as mandatory at the "B" and "A" levels and should be able to recognize the nature of mutually exclusive transactions (e.g. update, authorize, delete, add, etc.) that require different user authentication in support of separation of duty. This is a key systems feature which directly supports the meaningful implementation of the access control principle.

The enhancements proposed above for user authentication also are relevant to systems concerned with disclosure control, and we believe challenge-response authentication could reasonably be factored into any security system, regardless of particular security emphases. We make user authentication tokens required at division "B" because we believe that support for separation of duty is a minimal capability, and that a mechanism stronger than passwords is required for effective separation of duty.

There are several sorts of logging in the table. The simplest is a history log that records the identity of the user of the TP. This level of logging is effective if separation of duty is fixed in a static manner in the access control triples. A more powerful form of logging also records the user associated with each data modification, which provides a more detailed record of responsibility, and also supports that aspect of integrity that is based on the attribution of change.

Another form of logging is to record, for each data item, when IVPs have been executed for that data. This is a variant of a history log which may be separately retrievable, so that a user can determine the last time the integrity of the data was verified.

The most sophisticated form of logging, which remains rather speculative, is the labelling of data to indicate comparison with integrity domains.

The final feature is the system support for dynamic partition of TPs to support separation of duty. As was discussed in the previous section, this sort of functionality is probably very useful, but it is not clear how to build it into the system, as opposed to the application. For this reason, we indicated that it is desirable but do not suggest that it be required, even at the "A" division.

VI. Conclusion

We hope this paper clarifies some of the ideas the original paper left undeveloped. Still more work needs to be done in these areas. We believe future research should pay more attention to both the internal and external requirements for integrity. Future research also should focus on the implications of separation of duty, as we have only just started to understand the systems implications of this concept.

The Orange Book has been a very important start for setting industry security standards. Every reasonable attempt should be made to build on its structure. Because of its requirements, many tough problems--such as TP certification--are being tackled successfully. We believe that in the future the difficult problems with making and managing good logs and data labels will need to be addressed as well. Finally, confidentiality and integrity are only two pieces of the computer system security puzzle. The third piece, denial of service, needs to be addressed before we have a really complete approach.

Acknowledgements

The authors wish to thank Steve Lipner of Data Equipment Corporation and Bill Murray of Ernst & Whinney for advice and comments on earlier drafts of this paper. The authors also wish to thank Clark Holtzman of Ernst & Whinney for assistance in revising and editing the paper.

References

Andersen, Robert G., David D. Clark and David R. Wilson. "See-Through Security: A New Approach for Authenticating End Users in an Open Network." MISWeek, 1986.

[Canadian] Proceedings from the Canadian Trusted Computer Product Evaluation Criteria Workshop, August 4-5, 1988, Ottawa, Ontario. Govt. of Canada, 1988.

Clark, David D. and David R. Wilson. "A Comparison of Commercial and Military Security Policies." In Proceedings of the 1987 IEEE Symposium on Security and Privacy, April 27-29, 1987, Oakland, California. Pp. 184-94. Washington, D.C.: Computer Society Press of the IEEE, 1987.

[DoD] Department of Defense Trusted Computer System Evaluation Criteria, CSC-STD-011-83, Department of Defense Computer Security Center, Fort Meade, Md., August, 1983

Karger, Paul A. "Implementing Commercial Data Integrity with Secure Capabilities." In Proceedings of the 1988 IEEE Symposium on Security and Privacy, April 18-21, 1988, Oakland, California. Pp. 140-46. Washington, D.C.: Computer Society Press of the IEEE, 1988.

Lee, Theodore M.P. "Using Mandatory Integrity to Enforce 'Commercial' Security." In Proceedings of the 1988 IEEE Symposium on Security and Privacy, April 18-21, 1988, Oakland, California. Pp. 130-39. Washington, D.C.: Computer Society Press of the IEEE, 1988.

Murray, William H. "Data Integrity in a Business Data Processing System," Appendix 6. In Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS), October 27-29, 1987, Bentley College, Waltham, Massachusetts. Washington, D.C.: National Bureau of Standards, 1988.

[WIPCIS] Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS), October 27-29, 1987, Bentley College, Waltham, Massachusetts. Washington, D.C.: National Bureau of Standards, 1988.

Wiseman, Simon, et al. "The Trusted Path between SMITE and the User." In Proceedings of the 1988 IEEE Symposium on Security and Privacy, April 18-21, 1988, Oakland, California. Pp. 147-155. Washington, D.C.: Computer Society Press of the IEEE, 1988.

NCTL/NIST

GUIDE TO AUDITING FOR

CONTROLS AND SECURITY:

A SYSTEM DEVELOPMENT

LIFE CYCLE APPROACH

(NBS SP 500-153, APRIL, 1988)

ZELLA G. RUTHBERG

NATIONAL COMPUTER AND TELECOMMUNICATIONS LABORATORY

NATIONAL CENTER FOR STANDARDS AND TECHNOLOGY

BALTIMORE, MARYLAND

OCTOBER 20, 1988

NCTL/NIST

SYSTEM DEVELOPMENT LIFE CYCLE MATRIX

FRAMEWORK FOR SECURITY REVIEW/AUDIT

NCTL/NIST

I. BACKGROUND

A. WORK GROUP

B. MATRIX SOURCES

C. MATRIX SIGNIFICANCE

II. AIS LIFE CYCLE FUNCTIONAL MATRIX CONTENT

A. LIFE CYCLE PHASES

B. FUNCTIONAL ROLES

C. SOME EXAMPLES FROM MATRIX

D. DOCUMENTATION REQUIREMENTS

III. RELATION TO EDP AUDIT GUIDE

I.A.

PRESIDENT'S COUNCIL ON INTEGRITY AND EFFICIENCY (PCIE)

- FORMED MARCH 1981**

- CHAIR DEPUTY DIRECTOR OF THE OFFICE OF
 MANAGEMENT AND BUDGET**

- MEMBERS STATUTORY INSPECTORS GENERAL (IG'S)
 FEDERAL BUREAU OF INVESTIGATION
 DEPARTMENT OF JUSTICE
 OFFICE OF PERSONNEL MANAGEMENT**

- OBJECTIVES**
 - 1. INTERAGENCY PROGRAMS AND PROJECTS
 ON FRAUD AND WASTE**

 - 2. ENSURE SYSTEM INTEGRITY IN
 GOVERNMENT PROGRAMS AND OPERATIONS**

 - 3. OPTIMIZE USE OF RESOURCES ON
 CROSS-CUTTING ISSUES**

I.A.

PCIE WORK GROUP ON
EDP SYSTEMS REVIEW AND SECURITY

- FORMED OCTOBER 1983

- CHAIR RICHARD P. KUSSEROW

 IG OF THE DEPARTMENT OF HEALTH AND
 HUMAN SERVICES (HHS)

- PROJECT LEADER MS. BONNIE T. FISHER, HHS
 (UNTIL SEPT. 1985)

 MS. GAIL SHELTON, HHS
 (SEPT. '85 TO APRIL 1988)

- MEMBERS
 - o 14 FEDERAL DEPARTMENTS AND AGENCIES
 - HHS DOC GSA GAO
 - DOL NBS SBA EPA
 - HUD TREAS DOT NASA
 - DOD IRS DOE SMITHSONIAN
 - o IN IG, MANAGEMENT, TECHNICAL OFFICES

- GENERAL OBJECTIVES
 - o IMPROVE AUDIT OF AUTOMATED INFORMATION SYSTEMS (AIS'S) UNDER DEVELOPMENT AND OPERATIONAL
 - o INCREASE LIKELIHOOD OF AUDITABLE AND CONTROLLED SYSTEMS

I.B.

MAJOR SOURCE DOCUMENTS FOR MATRIX

- o DOD
 - DIRECTIVE 7910.1 (LIFE CYCLE CONCEPTS)
 - DIRECTIVE 7920.2 (LIFE CYCLE MANAGEMENT)
 - STRD 7935 (DOCUMENTATION STANDARDS)
- o NBS
 - FIPS PUBS 38, 64, 65, 73, 87, 101, 102
 - SPECIAL PUBS 500-98, 500-105
- o GAO
 - "YELLOW BOOK" (AUDIT STANDARDS)
- o OMB
 - A-123 (INTERNAL CONTROL)
 - A-71, TM1 (NOW A-130, ON SECURITY)
 - A-108 (NOW A-130, ON PRIVACY ACT)
 - A-127 (ON FINANCIAL MGMT)
- o GSA
 - FIRMR 201-30.007
 - 41 CFR 201-20
 - 41 CFR 201-32

I.C.

MATRIX SIGNIFICANCE

- o A FIRST CUT AT PROBLEM
 - ID OF SYSTEM LIFE CYCLE PHASES
 - WHO RESPONSIBLE FOR WHAT
- o APPLICABLE TO SOFTWARE & HARDWARE
- o REVIEWED & VALIDATED BY 76 FEDERAL AGENCIES
- o INTENDED FOR LARGE IN-HOUSE DEVELOPMENT EFFORTS
- o REQUIRES FLEXIBILITY FOR
 - SMALLER PROJECTS
 - CONTRACTOR DEVELOPMENT
 - PURCHASED SOFTWARE
- o CLARIFIES AUDIT ROLE
- o CLARIFIES SECURITY OFFICER ROLE
- o CLARIFIES QUALITY ASSURANCE ROLE
- o NEEDS TESTING BY AGENCIES

11.

FIGURE 1 AUTOMATED INFORMATION SYSTEM LIFE CYCLE
FUNCTIONAL MATRIX

LIFE CYCLE PHASES		OPERATIONAL ENVIRONMENT					
		I.	II.	III.	IV.	V.	VI.
PARTICIPANTS							
POLICY	1.						
AND	2.						
OVERSIGHT	3.						
	4.						
FUNCTIONAL	5.						
AND	6.						
OPERATIONAL	7.						
	8.						
	9.						

NOTE: TRANSITIONS BETWEEN PHASES REQUIRE APPROVAL AT HIGHEST LEVEL FOR EACH SYSTEM

LIFE-CYCLE PHASES			
PARTICIPANTS		OPERATING ENVIRONMENT	
POLICY / OVERSIGHT	Information Resources Management (IRM) Official [2]	<ul style="list-style-type: none"> ● establishes system life-cycle principles, documentation requirements, records management policy, & long-range system planning and decision process; with OIG establishes procedures by which OIG is notified of all significant new systems or systems modifications 	<ul style="list-style-type: none"> ● approves
	System Security Officer (SSO)/Internal Control Officer (ICO)	<ul style="list-style-type: none"> ● establishes Dept. internal control and computer security policy, per OMB Circulars A-123, A-127, & A-130; also develops policy pertaining to privacy requirements of agency records and data, per Privacy Act of 1974 	<ul style="list-style-type: none"> ● oversees evaluate
	Auditor (OIG)	<ul style="list-style-type: none"> ● develops ADP audit guide; conducts selective reviews or audits of automated systems, based on established criteria for prioritizing systems, and reports on needed management improvements [4] 	<ul style="list-style-type: none"> ● reviews/ Study, R: tem Dec: scope of
	Sponsor/User	<ul style="list-style-type: none"> ● establishes management level implementation guidelines & approval process for AISs; organizes a formal quality assurance function to provide for internal management reviews & recommendations pertaining to ADP efforts 	<ul style="list-style-type: none"> ● identifies ment; di: Cost/Ber Paper; sc
FUNCTIONAL / OPERATIONAL	Project Manager (PM)/Contracting Officer's Technical Representative (COTR) [5]	<ul style="list-style-type: none"> ● N/A 	<ul style="list-style-type: none"> ● develops Study, R: tem Dec:
	System Security Specialist (SSS)/Internal Control Specialist (ICS)	<ul style="list-style-type: none"> ● establishes policy implementation guidelines & planning processes for individual system development efforts, based on Dept., requirements and OMB Circulars A-123, A-127, & A-130 guidance 	<ul style="list-style-type: none"> ● conducts
	Contracting Officer/Contract Auditor [6]	<ul style="list-style-type: none"> ● establishes policy implementation guidelines based on GSA/Dept. procurement policy 	<ul style="list-style-type: none"> ● if appro: compliar
	ADP Manager	<ul style="list-style-type: none"> ● establishes technical policy implementation guidelines for in-house application development, purchased software & contracted system development efforts 	<ul style="list-style-type: none"> ● provides c: fice initial
	Quality Assurance (QA) Specialist	<ul style="list-style-type: none"> ● establishes and utilizes processes to insure applications systems meet requirements, including compliance with data processing procedures 	<ul style="list-style-type: none"> ● provides Statemer

[1] Matrix intended to reflect, primarily, roles & documents for large, in-house AIS development or redesign effort in body of report.

[2] IRM refers to "single official" as identified under PL96-511 and OMB Circular A-130. For smaller systems, not delegated, as provided for by Department policy.

[3] Relationship among IRM Official, Sponsor and ADP Manager may be formal, as in the case of an established body, depending upon the organization and particular system.

[4] All audit involvement in AIS life cycle should be based on an assessment of need and potential risk/exposure, all systems or phases.

[5] In some circumstances, some of these functions are handled by a COTR responsible to the Project Manager.

[6] In some circumstances, some of these functions are handled by a Contract Auditor responsible to the Contracting Officer.

Figure 1. AUTOMATED INFORMATION

I INITIATION	II DEFINITION	III SYSTEM DESIGN
<ul style="list-style-type: none"> ● approves Needs Statement ● oversees or conducts Risk Analysis; helps to evaluate system sensitivity ● reviews/evaluates Needs Statement, Feasibility Study, Risk Analysis, Cost/Benefit Analysis, and System Decision Paper; based upon review determines scope of future involvement ● identifies & validates need; develops Needs Statement; directs Feasibility Study, Risk Analysis, and Cost/Benefit Analysis; develops System Decision Paper; selects a Project Manager ● develops or oversees development of Feasibility Study, Risk Analysis, Cost/Benefit Analysis, and System Decision Paper ● conducts Risk Analysis as appropriate ● if appropriate, awards contract & assures contract compliance ● provides consultation as appropriate, unless this office initiates system ● provides consultation on quality attributes of Needs Statement 	<ul style="list-style-type: none"> ● approves System Decision Paper to advance to Phase II, in consultation with Sponsor/User and ADP Manager (occurs between Phases) [3] ● reviews SSO/ICO components of Project Plan, Functional Requirements Documents & Data Requirements Documents, on a select basis ● reviews/evaluates System Decision Paper, Project Plan, Functional Requirements Documents, Data Requirements Documents, and participates in their development, as necessary; prepares Audit Program ● approves Project Plan and Functional Requirements Documents, and updates System Decision Paper ● develops Project Plan and Functional and Data Requirements Documents with User participation ● provides consultation & review of SSO/ICO components of Project Plan, Functional Requirements Documents and Data Requirements Documents ● assures contract compliance ● reviews Project Plan, Functional Req. Doc's., Data Req. Doc's; as appropriate, provides technical support to Project Manager & Sponsor/User ● reviews project definition to ensure compliance with Needs Statement & data processing standards 	<ul style="list-style-type: none"> ● approves updated System Decision Paper to advance to Phase III, in consultation with Sponsor/User and ADP Manager (occurs between Phases), & enters system into Dept's. formal systems' inventory ● reviews SSO/ICO components of System/Subsystem, Program and Data Base Specifications, and Validation, Verification and Testing Plan and Specification ● reviews/evaluates & possibly inputs to Risk Analysis, Sys. Decision Paper, Sys./Subsys Program & Data Base Specs., VV&T Plan and Specs. and Revised Project Plan; updates Audit Program ● approves revised Project Plan and updates System Decision Paper; reassesses Risk Analysis; approves Validation, Verification and Testing Plan and Specifications (all based on QA recommendations) ● updates Project Plan; develops System/Subsystem Program & Data Base Specifications, & Validation, Verification and Testing Plan and Specifications ● reviews SSO/ICO components of System/Subsystem, Program and Data Base Specifications, and Validation, Verification and Testing Plan and Specification ● same as Phase II ● reviews VV&T components of Sys./Subsys., Prog. & Data Base Specs., & VV&T Plan and Specs; as appropriate, provides technical support to Project Manager and Sponsor/User in developing Specs. ● reviews system design, VV&T components and documentation for compliance to definition and data processing standards

redesign efforts. Alternative approaches are discussed systems, however, approval authorities commonly established AIS approval body, or informal ad hoc sk/exposure, and performed on a select basis, not on t Manager. the Contracting Officer.

SYSTEM (AIS) - LIFE-CYCLE MATRIX [1]

DEVELOPMENT

IV PROGRAMMING & TRAINING	V EVALUATION & ACCEPTANCE	VI INSTALLATION & OPERATION
<ul style="list-style-type: none"> ● approves updated System Decision Paper to advance to Phase IV, in consultation with Sponsor/User and ADP Manager (occurs between Phases) ● reviews SSO/ICO components of User Manual, Operations/Maintenance Manual, Installation and Conversion Plan, and revised VV&T Plan and Specifications ● reviews/evaluates revised Project Plan, System Decision Paper, revised VV&T Plan and Specifications, User Manual, Operations/Maintenance Manual, and Installation & Conversion Plan; updates Audit Program ● approves revised Project Plan, revised VV&T Plan and Specifications, User Manual, Operations/Maintenance Manual & Installation & Conversion Plan; updates System Decision Paper; initiates user training ● updates Project Plan; revises VV&T Plan and Specifications; develops User Manual, Operations/Maintenance Manual, and Installation & Conversion Plan; responsible for programming and testing ● reviews SSO/ICO components of User Manual, Operations/Maintenance Manual, Installation & Conversion Plan, and revised VV&T and Specifications Plan ● same as Phase II ● reviews VV&T components of User Manual, Operations/Maintenance Manual and Installation & Conversion Plan; provides technical support to Project Manager and Sponsor/User; may conduct DP training ● reviews program definition, program code, documentation, and training, for compliance to design and data processing standards 	<ul style="list-style-type: none"> ● approves updated System Decision Paper to advance to Phase V, in consultation with Sponsor/User and ADP Manager (occurs between Phases) ● reviews Test Analysis and Security Evaluation Report and SSO/ICO components of revised Installation & Conversion Plan ● reviews/evaluates revised Project Plan, revised Installation & Conversion Plan, and Test Analysis & Security Evaluation Report; updates Audit Program ● approves revised Project Plan and Installation & Conversion Plan; updates System Decision Paper; oversees training; accepts (accredits) system for operation ● updates Project Plan; supports & oversees Test Analysis & Security Eval. Report and certifies system security; revises User Manual, Operations/Maintenance Manual, and Installation and Conversion Plan based on test results ● reviews Test Analysis & Security Eval. Report and SSO/ICO impacted documentation updates to User Manual, Operations/Maintenance Manual, and Installation and Conversion Plan ● same as Phase II ● directs tests; reviews Test Analysis & Security Eval. Report, and Installation and Conversion Plan; continues to provide technical support; may do technical evaluation for certification. ● reviews Test Analysis & Security Eval. Report and advises responsible participants on system achievement of Needs Statement 	<ul style="list-style-type: none"> ● approves final installation of system; accredits systems determined to be of critical sensitivity (importance to the Dept.); directs periodic reviews per P.L. 96-511 for continued need ● conducts periodic reviews per OMB Circulars A-123, A-127, and A-130; feeding into long-range AIS planning process ● conducts periodic reviews per OMB A-130 & GAO audit standards; updates Audit Plan and Program as needed ● oversees training; directs periodic reviews of sensitive applications for recertification; identifies need for changes to system and revises Project Plan accordingly ● directs implementation and updates User Manual & Operations/Maintenance Manual as needed during implementation and operation ● conducts periodic reviews per OMB Circulars A-123, A-127, and A-130 ● if appropriate, continues to assure contract compliance ● conducts periodic reviews per OMB Circular A-130; provides technical assistance; maintains system documentation ● reviews changes to software system; summarize, analyzes and reports on defects to responsible participants

Key: The dot next to each entry in the matrix indicates whether that activity occurs within the phase or between two adjacent phases.

II.A.

MATRIX LIFE CYCLE SEGMENTS:

- o OPERATIONAL ENVIRONMENT
- o INITIATION -- PHASE I
- o DEFINITION -- PHASE II
- o SYSTEM DESIGN -- PHASE III
- o PROGRAMMING & TRAINING -- PHASE IV
- o EVALUATION & ACCEPTANCE -- PHASE V
- o INSTALLATION & OPERATION -- PHASE VI

II.A.

OPERATIONAL ENVIRONMENT

THE POLICIES AND GUIDELINES FOR LIFE CYCLE OF AN AUTOMATED INFORMATION SYSTEM

LIFE CYCLE PHASES

- o PHASE I -- INITIATION
 - PROBLEM RECOGNITION
 - IDENTIFICATION OF A NEED
 - VALIDATION OF THAT NEED
 - EXPLORATION OF ALTERNATIVE SOLUTIONS
 - COST/BENEFIT ANALYSIS
 - RISK ANALYSIS

SYSTEM DECISION (PAPER)

- o PHASE II -- DEFINITION
 - FUNCTIONAL REQUIREMENTS
 - DATA REQUIREMENTS
 - NEEDED INTERNAL CONTROLS
 - PROJECT PLAN
(INCLUDING: VERIFICATION, VALIDATION,
& TESTING (VV&T); CERTIFICATION &
ACCREDITATION)

II. A.

LIFE CYCLE PHASES (CONT'D)

UPDATE SYSTEM DECISION (PAPER)

- o PHASE III -- SYSTEM DESIGN
 - PROBLEM SOLUTION SPECIFICATIONS
 - INFORMATION AGGREGATES
 - INFORMATION FLOWS
 - LOGICAL PROCESSING STEPS
 - MAJOR INTERFACES
 - MAJOR INPUTS/OUTPUTS
 - SECURITY/INTERNAL CONTROL SPECS
 - PROJECT PLAN REVIEW
 - VV&T GOALS' IDENTIFICATION
 - CERTIFICATION/ACCREDITATION PLAN REVIEW

UPDATE SYSTEM DECISION (PAPER)

- o PHASE IV -- PROGRAMMING & TRAINING
 - IMPLEMENT DESIGN INTO CODE
 - TRAIN IN USE OF SYSTEM
 - PREPARE SYSTEM MANUALS
 - USERS' ; O & M
 - PREPARE PRELIMINARY INSTALLATION PLAN
 - PROJECT PLAN REVIEW

II. A.

LIFE CYCLE PHASES (CONT'D)

UPDATE SYSTEM DECISION (PAPER)

o PHASE V -- EVALUATION & ACCEPTANCE

- EXECUTE VV&T PLAN
- FIELD TEST
- CERTIFY SENSITIVE SYSTEMS
- ACCREDIT SENSITIVE SYSTEMS
- REVISE MANUALS AS NEEDED
- REVISE INSTALLATION PLAN

APPROVE SYSTEM INSTALLATION

o PHASE VI -- INSTALLATION & OPERATION

- IMPLEMENT OPERATIONAL PLAN
(E.G., EXTENSION TO OTHER SITES)
- CONTINUE APPROVED OPERATIONS
- BUDGET ADEQUATELY
- MAINTAIN THE AIS WITH CHANGE CONTROL
- PERIODIC REVIEWS FOR INTERNAL CONTROLS &
SECURITY (RECERTIFICATION)

11.B.

PARTICIPANTS IN LIFE CYCLE

- o POLICY & OVERSIGHT
 - INFORMATION RESOURCE MANAGER
 - SYSTEM SECURITY OFFICER/
INTERNAL CONTROL OFFICER
 - AUDITOR
 - SPONSOR/USER
- o FUNCTIONAL & OPERATIONAL
 - PROJECT MANAGER/COTR
 - SYSTEMS SECURITY SPECIALIST/
INTERNAL CONTROL SPECIALIST
 - CONTRACTING OFFICER/
CONTRACT AUDITOR
 - ADP MANAGER
 - QUALITY ASSURANCE SPECIALIST

11.B.

PARTICIPANTS -- POLICY & OVERSIGHT

1. INFORMATION RESOURCE MANAGER (IRM)

(CALLED FOR IN PL96-511 & OMB A-130)

- o DEVELOPS UNIFORM POLICIES & PROCEDURES
- o MANAGES RECORDS/INFORMATION
- o MANAGES INFORMATION RESOURCES
- o APPROVES DEVELOPMENT/ACQUISITION

2A. SYSTEM SECURITY OFFICER (SSO) (A-130)

- o DESIGNATED BY IRM
- o DEVELOPS, IMPLEMENTS, OPERATES
COMPUTER SECURITY PROGRAM
- o DEFINES & APPROVES SYSTEM SECURITY SPECS FOR
NEW OR CHANGED SYSTEM,
IN-HOUSE OR ACQUIRED SYSTEM
- o OVERSEES RISK ANALYSIS DURING
SYSTEM LIFE CYCLE

2B. INTERNAL CONTROL OFFICER (ICO) (A-123), FMFIA

- o OVERSEES FINANCIAL MANAGEMENT & INFO SYSTEMS
(IDENTIFICATION, DEVELOPMENT, MAINTENANCE,
REVIEW, IMPROVEMENT)
- o OVERSEES VULNERABILITY ASSESSMENTS
- o ESTABLISHES POLICY WITH RESPECT TO
CONTROL POINTS

IIB.

PARTICIPANTS -- POLICY & OVERSIGHT (CONT'D)

3. AUDITOR (OIG) ("YELLOW BOOK")

- o **REVIEWS AUTOMATED INFORMATION SYSTEMS
DEVELOPMENTAL
OPERATIONAL**
- o **LOOKS FOR COMPLIANCE WITH
REQUIREMENTS
POLICIES
PROCEDURES
DURING SYSTEM DEVELOPMENT**
- o **ASSURES THROUGHOUT SYSTEM LIFE CYCLE
PROPER INTERNAL CONTROL
AUDITABILITY
SECURITY NEEDS SATISFIED**
- o **REVIEWS ACQUISITION STRATEGY FOR
INTERNAL CONTROL
AUDITABILITY
SECURITY NEEDS**

IIB.

PARTICIPANTS -- POLICY & OVERSIGHT (CONT'D)

4. SPONSOR/USER

- IDENTIFIES AIS NEED
- IDENTIFIES ALTERNATIVE SOLUTIONS
- DETERMINES FEASIBILITY
- DETERMINES COST/BENEFIT
- OVERSEES RISK ANALYSIS
- ACCREDITS SYSTEM
- RESPONSIBLE FOR SUCCESS/FAILURE OF SYSTEM
- MAY SERVE ON SYSTEM APPROVAL/REVIEW BOARD

IIB.

PARTICIPANTS -- FUNCTIONAL & OPERATIONAL

5. PROJECT MANAGER (PM)/COTR

- DESIGNATED BY SPONSOR/USER
- RESPONSIBLE TO SPONSOR/USER
- ASSURES
 - PROPER SYSTEM DESIGN
 - DEVELOPMENT ON SCHEDULE
 - SYSTEM DOCUMENTATION PREPARED
- OVERSEES CERTIFICATION OF TECHNICAL SPECS

6A. SYSTEM SECURITY SPECIALIST (A-130)

- ASSURES COMPLIANCE WITH AGENCY SECURITY POLICY PRIOR TO INSTALLATION
- APPROVES DESIGN REVIEWS
 - DESIGN MEETS SECURITY SPECS
 - DESIGN MEETS SYSTEM TESTS
- COORDINATES WITH A-123 & A-127 REQUIREMENTS

IIB.

PARTICIPANTS -- FUNCTIONAL & OPERATIONAL (CONT'D)

6B. INTERNAL CONTROL SPECIALIST (A-123)

- o ASSURES COMPLIANCE WITH INTERNAL CONTROL POLICY
- o ASSURES SYSTEM MEETS BASIC STANDARDS FOR
 - DOCUMENTATION
 - RECORDING OF TRANSACTIONS
 - EXECUTION OF TRANSACTIONS
 - SEPARATION OF DUTIES
 - ACCESS TO RESOURCES, ETC.

7A. CONTRACTING OFFICER

- o AWARDS & MANAGES DEVELOPMENT AND/OR PROCUREMENT CONTRACT
- o PROCURES NEW APPLICATION SOFTWARE
- o WORKS WITH PM AND SPONSOR ON RFP

7B. CONTRACT AUDITOR

- o REVIEWS CONTRACTOR PERFORMANCE
(ON REQUEST)

IIB.

PARTICIPANTS -- FUNCTIONAL & OPERATIONAL (CONT'D)

8. ADP MANAGER

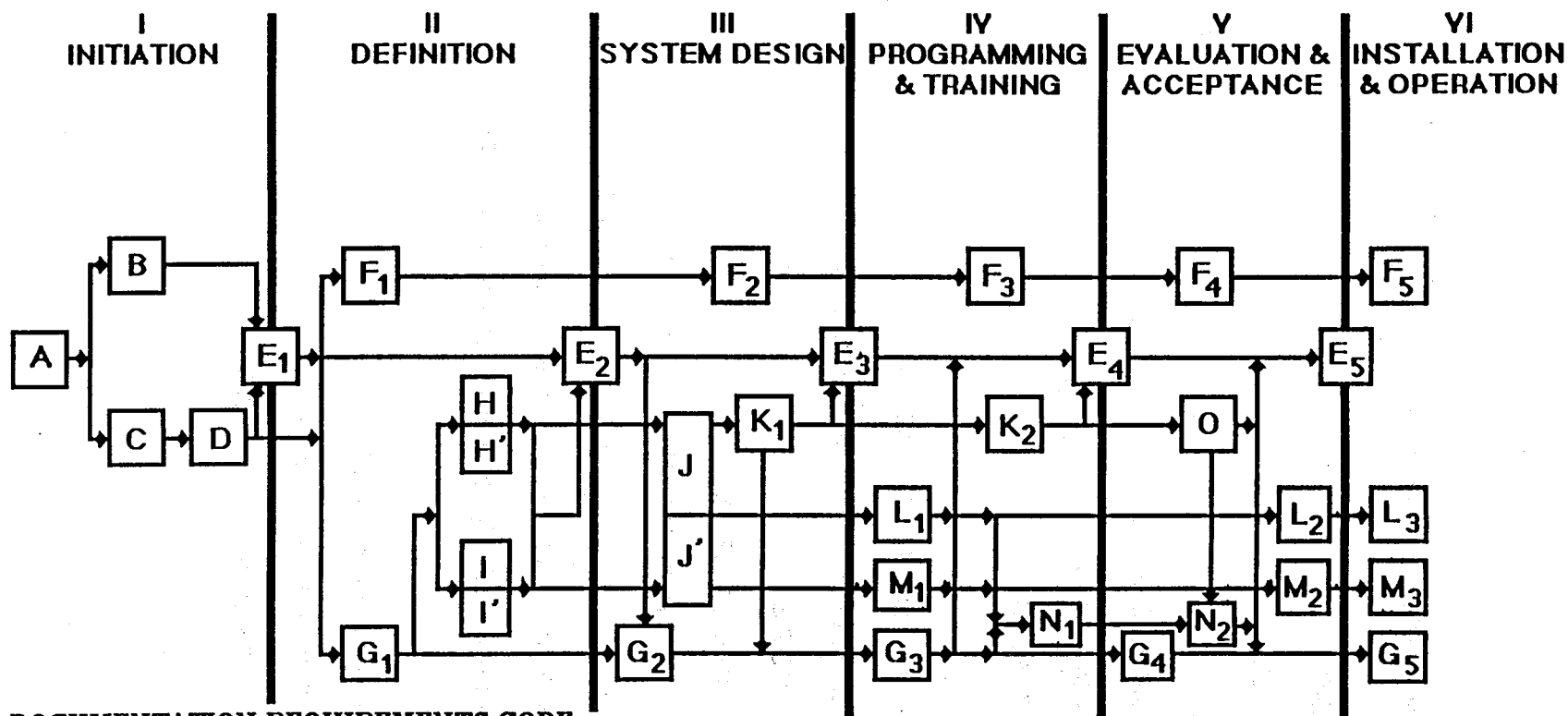
- o MANAGES ADP INSTALLATIONS
- o MANAGES OPERATIONS OF ADP PROGRAMS
- o MAY DO DEVELOPMENT ON AIS
- o MAY SERVE ON SYSTEM REVIEW/APPROVAL BOARD
- o MAY INITIATE A DEVELOPMENT EFFORT
- o MAY BE RESPONSIBLE TECHNICAL OFFICIAL FOR CERTIFICATION

9. QUALITY ASSURANCE SPECIALIST (QA)

ASSURES THAT:

- o SYSTEM SATISFIES OBJECTIVES
- o SYSTEM CONTAINS INTERNAL CONTROLS FOR RELIABILITY
- o SYSTEM CONFORMS WITH DP PROCEDURES & REQUIREMENTS
- o PERFORMS INDEPENDENT REVIEWS
- o COORDINATES WITH SECURITY/AUDIT

Figure 2. SYSTEM LIFE-CYCLE DOCUMENTATION FLOW CHART



DOCUMENTATION REQUIREMENTS CODE

- A. Needs Statement (FIPS PUB 64, DOD 7920.2)*
- B. Feasibility Study (FIPS PUB 64)*
- C. Risk Analysis (FIPS PUB 65)*
- D. Cost/Benefit Analysis (FIPS PUB 64)*
- E. System Decision Paper (FIPS PUB 64, DOD 7920.2)*
- F. Audit Plan
- G. Project Plan (FIPS PUB 102 & 105, NBS SP 500-98)*
- H. Functional Requirements Document (FIPS PUB 38 & 124)*
- H'. Functional Security and Internal Control Requirements Document (FIPS PUB 73 & 102)*

- I. Data Requirements Document (FIPS PUB 38)*
- I'. Data Sensitivity/Criticality Description (FIPS PUB 65 & 102)*
- J. System/Sub System, Program & Data Base Specifications (FIPS PUB 38)*
- J'. Security and Internal Control Related Specifications (FIPS PUB 73 & 102)*
- K. Validation, Verification and Testing Plan and Specifications (FIPS PUB 101)*
- L. User Manual (FIPS PUB 38, NBS SP 500-98)*
- M. Operations/Maintenance Manual (FIPS PUB 38, FIPS PUB 106, NBS SP 500-98)*
- N. Installation & Conversion Plan (FIPS PUB 101, NBS SP 500-105)*
- O. Test Analysis & Security Evaluation Report (FIPS PUB 38 & 102, NBS SP 500-98)*

* Source

(Note: Document subscripts refer to successive iterations of that document.)

II.D.

DOCUMENTATION FLOW CHART DOCUMENTS

- A. NEEDS STATEMENT**
- B. FEASIBILITY STUDY**
- C. RISK ANALYSIS**
- D. COST/BENEFIT ANALYSIS**
- E. SYSTEM DECISION PAPER**
- F. INTERNAL AUDIT PLAN**
- G. PROJECT PLAN**
- H. FUNCTIONAL REQUIREMENTS DOCUMENT**
- H'. FUNCTIONAL SECURITY & INTERNAL CONTROL REQUIREMENTS DOCUMENT**
- I. DATA REQUIREMENTS DOCUMENT**
- I'. DATA SENSITIVITY/CRITICALITY DESCRIPTION**
- J. SYSTEM/SUBSYSTEM, PROGRAM, & DATA BASE SPECS**
- J'. SECURITY & INTERNAL CONTROL RELATED SPECS**
- K. VALIDATION, VERIFICATION & TESTING PLAN & SPECS**
- L. USERS' MANUAL**
- M. OPERATIONS MAINTENANCE MANUAL**
- N. INSTALLATION & CONVERSION PLAN**
- O. TEST RESULTS & EVALUATION REPORT**

II.D.

DOCUMENTATION FLOW CHART

o COMMENTS

- SUBSCRIPTS INDICATE UPDATES TO DOCUMENTS WHICH ARE REVISED ITERATIVELY
- PROJECT PLAN UPDATED AT BEGINNING OF EACH PHASE
- SELECT DOCUMENTS FLOW INTO OTHER DOCUMENTS
- FLOW CHART IS A GUIDANCE DOCUMENT
- DOCUMENTATION MANAGEMENT PROGRAM NEEDED EARLY IN SYSTEM LIFE CYCLE
- DOCUMENTS REFERENCED ARE DELIVERABLES IN EACH PHASE
- FLEXIBLE USE OF THIS GUIDANCE IS NEEDED.

FOR EXAMPLE:

1. VV&T MIGHT BE INTEGRATED IN PROJECT PLAN.
2. FEASIBILITY STUDY, COST/BENEFIT ANALYSIS, AND RISK ANALYSIS MIGHT BE INTEGRATED INTO SYSTEM DECISION PAPER.

II.C.

EXAMPLE 1 -- LIFE CYCLE FUNCTIONAL ROLE OF AUDITOR (OIG)

OPERATING ENVIRONMENT

- o DEVELOPS EDP AUDIT GUIDE
- o CONDUCTS SELECTED REVIEWS OF AIS'S
- o PRIORITIZES ACTIVITIES
- o REPORTS TO MANAGEMENT ON NEEDED
IMPROVEMENTS

I. INITIATION

- o SELECTIVELY REVIEWS/EVALUATES
 - NEEDS STATEMENT
 - FEASIBILITY STUDY
 - RISK ANALYSIS
 - COST/BENEFIT ANALYSIS
 - SYSTEM DECISION PAPER
- o DETERMINES SCOPE OF FUTURE WORK

II.C.

EXAMPLE I -- LIFE CYCLE FUNCTIONAL ROLE OF
AUDITOR (OIG) (CONT'D)

II. DEFINITION

- o SELECTIVELY REVIEWS/EVALUATES
 - SYSTEM DECISION PAPER
 - PROJECT PLAN
 - FUNCTIONAL REQUIREMENTS DOCUMENTS
 - DATA REQUIREMENTS DOCUMENTS
- o PARTICIPATES SELECTIVELY IN DEVELOPMENT
- o PREPARES AUDIT PROGRAM

III. SYSTEM DESIGN

- o SELECTIVELY REVIEWS/EVALUATES &
POSSIBLY INPUTS TO
 - RISK ANALYSIS
 - SYSTEM DECISION PAPER
 - SYSTEM, PROGRAM, DATA BASE SPECS
 - VV&T PLAN AND SPECIFICATIONS
 - REVISED PROJECT PLAN
- o UPDATES AUDIT PROGRAM

II.C.

EXAMPLE I -- LIFE CYCLE FUNCTIONAL ROLE OF
AUDITOR (OIG) (CONT'D)

IV. PROGRAMMING & TRAINING

- o SELECTIVELY REVIEWS/EVALUATES
 - REVISED PROJECT PLAN
 - REVISED VV&T PLAN AND SPECS
 - USERS' MANUAL
 - OPERATIONS & MAINTENANCE MANUAL
 - INSTALLATION & CONVERSION PLAN
- o UPDATES INTERNAL AUDIT PROGRAM

V. EVALUATION & ACCEPTANCE

- o SELECTIVELY REVIEWS/EVALUATES
 - REVISED PROJECT PLAN
 - REVISED INSTALLATION & CONVERSION PLAN
 - TEST ANALYSIS & SECURITY
- o EVALUATION REPORT
UPDATES AUDIT PROGRAM

VI. INSTALLATION & OPERATION

- o CONDUCTS PERIODIC REVIEWS
(A-130 & GAO AUDIT STANDARDS)
- o UPDATES AUDIT PLAN
AND AUDIT PROGRAM (AS NEEDED)

IIC.

EXAMPLE II -- LIFE CYCLE FUNCTIONAL ROLE OF
SYSTEM SECURITY/INTERNAL CONTROL
OFFICER (SSO/ICO)

OPERATING ENVIRONMENT

- o ESTABLISHES COMPUTER SECURITY POLICY
PER A-123, A-127, AND A-130
- o DEVELOPS PRIVACY POLICY PER
PRIVACY ACT OF 1974

1. INITIATION

- o OVERSEES OR CONDUCTS RISK ANALYSIS
- o HELPS EVALUATE SYSTEM SENSITIVITY

2. DEFINITION

- o SELECTIVELY REVIEWS SSO/ICO COMPONENTS OF
PROJECT PLAN
FUNCTIONAL REQUIREMENTS DOCUMENTS
DATA REQUIREMENTS DOCUMENTS

3. SYSTEM DESIGN

- o REVIEWS SSO/ICO COMPONENTS OF
SYSTEM/SUBSYSTEM
PROGRAM & DATA BASE SPECS
V V & T PLAN & SPECS

IIC.

EXAMPLE 11 -- LIFE CYCLE FUNCTIONAL ROLE OF
SYSTEM SECURITY/INTERNAL CONTROL
OFFICER (SSO/ICO) (CONT'D)

4. PROGRAMMING & TRAINING

- o REVIEWS SSO/ICO COMPONENTS OF
USER MANUAL
O & M MANUAL
INSTALLATION & CONVERSION PLAN
REVISED V V & T PLAN & SPECS

5. EVALUATION & ACCEPTANCE

- o REVIEWS
TEST ANALYSIS & SECURITY
EVALUATION REPORT
SSO/ICO COMPONENTS OF REVISED
INSTALLATION & CONVERSION PLAN

6. INSTALLATION & OPERATION

- o CONDUCTS PERIODIC REVIEWS PER
A-123, A-127, A-130
- o FEEDS INFORMATION INTO LONG RANGE
AIS PLANNING PROCESS

II.C.

EXAMPLE III -- LIFE CYCLE FUNCTIONAL ROLE OF
QUALITY ASSURANCE SPECIALIST

OPERATIONAL ENVIRONMENT

- o ESTABLISHES & USES PROCESSES TO INSURE
APPLICATION SYSTEM MEETS REQUIREMENTS
- o CHECKS COMPLIANCE WITH DP PROCEDURES

I. INITIATION

- o CONSULTS ON QUALITY ATTRIBUTES OF
NEEDS STATEMENT

II. DEFINITION

- o REVIEWS PROJECT DEFINITION FOR
 - COMPLIANCE WITH NEEDS STATEMENT
 - COMPLIANCE WITH DP STANDARDS

III. SYSTEM DESIGN

- o REVIEWS FOR COMPLIANCE TO PROJECT
DEFINITION & DATA PROCESSING STANDARDS
 - SYSTEM DESIGN
 - V V & T COMPONENTS
 - DOCUMENTATION

II.C.

EXAMPLE III -- LIFE CYCLE FUNCTIONAL ROLE OF
QUALITY ASSURANCE SPECIALIST (CONT'D)

IV. PROGRAMMING & TRAINING

- o REVIEWS FOR COMPLIANCE TO DESIGN &
DATA PROCESSING STANDARDS
 - PROGRAM DEFINITION
 - PROGRAM CODE
 - DOCUMENTATION
 - TRAINING

V. EVALUATION & ACCEPTANCE

- o REVIEWS TEST ANALYSIS & SECURITY
EVALUATION REPORT
- o ADVISES ON SYSTEM ACHIEVEMENT OF
NEEDS STATEMENT

VI. INSTALLATION & OPERATION

- o REVIEWS CHANGES TO SOFTWARE SYSTEM
- o SUMMARIZES, ANALYZES, REPORTS ON DEFECTS

NCTL/NIST

AIS DEVELOPMENTAL AUDITS/REVIEWS

1. REVIEW OF SDLC METHODOLOGY
2. CONTROL OBJECTIVES
3. APPROACH
4. AUDIT/REVIEW PROGRAM FOR EACH PHASE

NCTL/NIST

CATEGORIES OF CONTROLS

1. GENERAL

A. MANAGEMENT

B. ENVIRONMENTAL

2. APPLICATION

NCTL/NIST

CONTROL OBJECTIVES/CONCERNS

(FROM GAO "YELLOW BOOK")

- 1. LEGAL REQUIREMENTS**
- 2. MANAGEMENT POLICIES**
- 3. INTERNAL CONTROLS**
- 4. AUDIT TRAILS**
- 5. DOCUMENTATION**
- 6. ECONOMY & EFFICIENCY**

NCTL/NIST

SDLC METHODOLOGY REVIEW
(FOR EACH PHASE)

- I. BECOME FAMILIAR WITH ORGANIZATION'S
SYSTEM DEVELOPMENT METHODOLOGY (SDM)
 1. PRIOR EVALUATION OF SDM?
 2. UP TO DATE?
 3. USED?
 4. KNOWN PROBLEMS?
 5. PERMITTED DEVIATIONS?
 6. FORMALLY STRUCTURED INTO PHASES?
 7. FORMAL DOCUMENTATION REQUIREMENTS?
 8. EMPHASIS ON SECURITY AND INTERNAL CONTROL?

SDLC METHODOLOGY REVIEW

(FOR EACH PHASE)

I. BECOME FAMILIAR WITH ORGANIZATION'S SDM (CONT'D)

9. PLANNING REQUIREMENT FOR EACH PHASE?
10. CHANGE CONTROL OVER REQUIREMENTS?
11. ADP STEERING COMMITTEE TO MANAGE SDM?
12. FORMAL RECOGNITION OF FUNCTIONAL ROLES?
13. SUFFICIENT COMMUNICATION BETWEEN USERS & DEVELOPERS?
14. FORMAL DOCUMENTATION REQUIREMENTS?
15. USE OF WELL-DEFINED STANDARDS?
16. SUFFICIENT SUPPORT BY TOP MANAGEMENT?

NCTL/NIST

SDLC METHODOLOGY REVIEW

(FOR EACH PHASE - CONT'D)

II. COMPARE ORGANIZATION'S SDLC METHODOLOGY TO
AUDIT GUIDE MATRIX, WITH RESPECT TO

1. REQUIRED DOCUMENTS
2. MISSING DOCUMENTS OR PARTS OF DOCUMENTS
3. IMPORTANCE OF MISSING DOCUMENTS

NCTL/NIST

**APPROACH
(FOR EACH PHASE)**

- 1. AUDIT/REVIEW LEVEL**
- 2. PRIMARY AUDIT/REVIEW OBJECTIVE**
- 3. OVERVIEW OF PHASE**
- 4. AUDIT/REVIEW SURVEY**
- 5. CUSTOMIZED AUDIT/REVIEW OBJECTIVES**
- 6. DETAILED AUDIT/REVIEW TESTING
(USING TEST TABLES)**
- 7. ASSESSMENT OF AUDIT/REVIEW RESULTS**
- 8. REASSESS AUDIT/REVIEW STRATEGY**

AUDIT TEST TABLES

#	AUDIT OBJECTIVES/ INDICATORS	AUDIT TEST	TOOLS AND TECHNIQUES

TABLE 4.3 - SYSTEM DESIGN PHASE AUDIT TESTS

#	AUDIT OBJECTIVES/ INDICATOR	AUDIT TEST	TOOLS AND TECHNIQUES
1.	The revised Project Plan is current and provides the direction needed to effectively and efficiently manage the project.	Confirm with the Project Manager that the plan is up to date, is being followed, and provides adequate information to adjust project direction as appropriate to ensure the project will be completed on time, within budget, and produce the expected deliverables.	<p>Compare the status of completed documents to document status as included in the Project Plan.</p> <p>Verify that the plan is accurate, and then, through interview with the document developers, ensure that problems in their work are appropriately addressed by project management.</p>
2.	The final system design should be approved by all appropriate levels of management as meeting all predetermined needs.	<p>Determine that user department management, and other appropriate management, have reviewed the system design specifications/documents.</p> <p>Confirm that user department management(s) has approved the design as meeting their needs.</p>	<p>Manual examination of evidence indicating that the material has been reviewed (e.g., reviewing minutes of meeting, department memorandum, departmental time sheets, etc.).</p> <p>Examine user "signoff" of design phase documents.</p> <p>System development scheduling software -- Obtain status information from the scheduling packages.</p>

TABLE 4.3 - SYSTEM DESIGN PHASE AUDIT TESTS

#	AUDIT OBJECTIVES/ INDICATOR	AUDIT TEST	TOOLS AND TECHNIQUES
3.	Sufficient data processing and security controls should be incorporated in the detailed design to ensure the integrity of the system.	<p>Review the detailed design specifications and identify the system controls to be built in the system to evaluate the adequacy of those controls. (Note: If control documentation does not exist within the system design documents, this can be an extremely time-consuming task for the auditor.)</p> <p>The areas to be addressed are:</p> <ol style="list-style-type: none"> 1. How are the controls specified in requirements designed into system? 2. What new risks does design introduce and what controls reduce risk? 3. What mechanisms have been designed to ensure ongoing integrity (e.g., exception reports, control total comparisons)? 4. What are the access control mechanisms? 	<p>Risk points are where controls should be placed. The auditor has a variety of strategies available to identify control points. The adequacy of controls should be addressed at those points. If the system has been designed using structured design, then the nodes in the structure indicate the points where controls should be exercised. The auditor can use the structured design to show the data flow in the points where data should be controlled. The controls can be documented on the structured design, with the absence of control at the nodes of the structure indicating potential control weakness.</p> <p>If controls are not documented, and the system is not designed using a structured method, then the auditor has the option of selecting one or more of the control design methodologies available either through the private or public sector. Among the most common are:</p> <ul style="list-style-type: none"> ● "Black Book" [GAO81-3] -- issued by the GAO

TABLE 4.3 - SYSTEM DESIGN PHASE AUDIT TESTS

#	AUDIT OBJECTIVES/ INDICATOR	AUDIT TEST	TOOLS AND TECHNIQUES
3.	(continued)		<ul style="list-style-type: none"> ● <u>Auditing Computer Applications</u> -- issued by AUERBACH and based on the GAO "Black Book" ● Control matrix analysis -- available through Touche Ross & Co. and described in their book <u>Computer Control and Audit</u> by Mair, Wood, and Davis ● Transaction flow auditing -- materials available through Arthur Andersen and Company <p>If emphasis is on the security part of control, then FIPS PUB 65 should be referenced as a security assessment methodology, and NBS SP 500-133 for other methods.</p>
4.	Rules for authorizing transactions should be defined and documented.	Determine that the method for authorizing each transaction has been documented, and that the method is reasonable. (Note that the audit process for this will vary depending on whether the transaction originates on paper, or whether it originates electronically.)	For paper transactions, use a structured interview technique to validate that all transactions have been identified, and that the rules for authorizing those transactions are defined. Note that, for financial systems, the financial officer of the agency should be the individual indicating how financial transactions are authorized.

C3-CR01-88
17 October 1988

Library No. S-231, 209

THE LOCK DEMONSTRATION

**CONSTRUCTING AN INFOSEC SYSTEM USING
LOCK TECHNOLOGY**

W.E. Boebert/Honeywell Secure Computing
Technology Center

**UPDATE PROCESSING IN LDV: A SECURE
DATABASE SYSTEM**

Paul Stachour, Bhavani Thuraisingham,
Patricia Dwyer

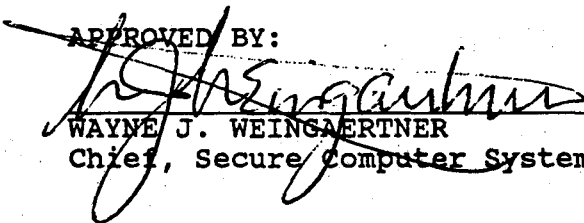
CONTRACT SPONSORS: Office of Research and Development
National Computer Security Center

Rome Air Development Command

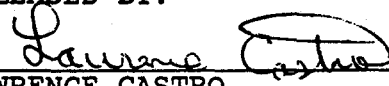
CONTRACT NOS: MDA-904-87-C-6011
F30602-86-C-0003

This report, C3 CR01-88, has been reviewed by the Office of
Research and Development and is approved for publication.

APPROVED BY:


WAYNE J. WEINGAERTNER
Chief, Secure Computer Systems Division

APPROVED AND
RELEASED BY:


LAWRENCE CASTRO
Chief, Office of Research and Development

C3-CR01-88
17 October 1988

Companies seeking additional information should write to:

National Computer Security Center
ATTN: C31, LOCK
9800 Savage Road
Ft. George G. Meade, MD 20755-6000

Honeywell Secure Computing Technology Center
ATTN: LOCK
2855 Anthony Ln. S.
Minneapolis, MN 55418

The LOCK Demonstration

W.E. Boebert
Honeywell
Secure Computing Technology Center
2855 Anthony Lane S.
Minneapolis MN 55418

Abstract

The LOCK project intends to demonstrate the proof of principle of innovative approaches to technology, policy and doctrine by integrating these elements into a prototype system. There will be technical demonstrations of this prototype 24 and 42 months into the project. This paper describes the larger demonstration context in which these technical demonstrations reside and provides an initial description of the demonstration to take place at 24 months.

1. THE LOCK APPROACH

The LOCK project [1] takes an approach to the development and fielding of Trusted Computing Bases (TCBs) at the higher rating levels that contrasts sharply with current practice. The various architectural and policy innovations of LOCK were, in many cases, motivated by this distinct doctrinal approach.

1.1 Traditional TCB Development

The pre-LOCK approach traces its roots back to the formation of the Computer Security Initiative over a decade ago. The fruits of that initiative were the NBS Conferences, the Trusted Computer System Evaluation Criteria (TCSEC), and the formation of the National Computer Security Center as overseer of the Evaluated Product List. These efforts were implicitly based on the notion that computer security was primarily an exercise in restructuring operating systems, and that manufacturers would be motivated to perform that restructuring by the increased business gained from having a product on the Evaluated Product List.

1.2 Problems With The Traditional Approach

The current approach has worked well up to the C2/B1 rating level. These ratings can, in general, be achieved by software cleanup or modification. A "market pull," especially at the C2 level, has been provided by mandating that all U.S. Government computers be C2 or above by a specified date. One of the important factors in the success of the policy at the lower rating levels has been the ability of independent vendors to obtain certification of add-on packages.

The situation is sharply different at the higher rating levels, where the Evaluated Product List is very sparsely populated. Several factors have contributed to this situation:

1. The problem of achieving higher rating levels is technically much more difficult and the evaluation process is longer; evaluation of a mature product could consume much if not all of its remaining market life.

2. The lack of worked examples or any other technology transfer vehicle raises the cost of development. This latter factor is made worse by the proprietary nature of evaluated technology, which prevents any cross-vendor feedback.

3. The original notion of a Trusted Computing Base was devised in the days of "full-range" computer vendors who provided their customers with a supported spectrum of hardware, operating systems, and applications. Computing power was delivered primarily by large, stand-alone mainframes. Multiple-vendor sites were the minority, especially in the Government. It seemed natural to view security as a feature which such vendors would incorporate in their product lines, especially if a "market pull" existed as a consequence of mandated rating levels for procurements.

Since then, the number of "full-range" vendors has shrunk and their market share has diminished. Computing power is now delivered by smaller and more specialized vendors who have much shorter product cycles, little if any existing operating system technology to use as a base for a secure system, and a need to invest in technology that will bring a rapid return. Such vendors are reluctant to devote the resources required to develop, from scratch, systems at the higher rating levels. Very little operating system development has been done and even less is going on now.

The corresponding lack of available products leads procurement authorities to be very reluctant to mandate rating levels greater than C2. Thus the Infosec effort at those levels remains in the circular wait that has existed since the early 1970's: procurement authorities will not mandate rating levels for which no products exist and vendors will not develop products for which no market exists.

4. The existing view of a TCB as a free-standing, monolithic entity which provides operating systems services in a secure fashion has been shown to have technical shortcomings as well as the doctrinal problems described above. In particular, attempts to construct practical TCBs at the higher assurance levels has shown that application-dependent, security-relevant software is substantially more extensive and important than anyone thought in the early days.

This situation significantly reduces the relevance and therefore the marketability of any TCB which is evaluated outside the context of a specific application. The result is a "double bind" for the effort to populate the Evaluated Product List at the higher ratings. If the evaluation process includes application-dependent functionality, then the duration of the evaluation will be increased to the point where those applications will have evolved beyond recognition by the time the TCB is certified. If application-dependent functionality is ignored, the resulting TCB will provide little more than bare-bones operating system services and be unattractive in the marketplace.

5. The integration of computing and communications, and the corresponding requirement to integrate cryptography and computer security techniques at the higher rating levels, has dealt the final blow to the idea of an Infosec TCB as a special operating system which runs on a standard machine. Endorsed cryptography requires specialized approaches to architecture, implementation, and packaging, and is therefore heavily involved with the design of the hardware.

1.3 The LOCK Approach

The LOCK project deliberately took on the problem of delivering an Infosec capability into an environment of rapid technological change, vendor profusion and turnover, and security-relevant applications. The project embraced the paradigm for technology infusion which was demonstrated by the IBM PC and "clone" experience, with its explosion of specialized vendors, rather than the mainframe vendor paradigm which influenced earlier Compuserc doctrine.

The LOCK architecture supports this decentralized paradigm for technology infusion in three important ways:

1.3.1 Hardware Security Modules—LOCK isolates both the hardcore computer security function (the reference monitor) and the cryptography in sealed hardware modules which behave essentially like peripherals. The hardware reference monitor module is called the SIDEARM and the cryptographic module is called the Bulk Encryption Device, or BED. If the LOCK project succeeds, it should be possible for hardware vendors to incorporate these modules into architectures at significantly less cost, risk, and schedule than the development of an operating system to A1 or greater criteria.

Physical isolation of critical Infosec functions increases assurance and decouples the implementation of these functions from any specific processor. This decoupling reduces the degree to which the security functions depend on the correct operation of a given processor, and opens up the possibility that a single set of modules can be incorporated into many different machines. Physical isolation and encapsulation also reduces the amount of computer security and cryptographic expertise required to incorporate these functions in the machine, an important consideration for vendors who do not wish to deal with classified information. Finally, the isolation of these functions into special hardware means that the design and especially the assurance costs of that hardware can be spread over several generations of several machines.

1.3.2 Open Architecture—The security policy implemented by SIDEARM is distinct from predecessor policies in that it incorporates rules for the enforcement of software structure; these rules, called "Type Enforcement," [2] are in addition to the Mandatory and Discretionary policies required by the TCSEC. Type Enforcement represents a significant advance in practical security policies in that it enables proofs of security subsystems to be "factored" into orderly demonstrations of global or structural properties (e.g., unby-passability) and local or detailed properties such as the correspondence of module code to its specification. This factoring mirrors, on the assurance side, the distinction between "programming in the large" and "programming in the small" in contemporary implementation methodologies.

Type Enforcement is used to maintain the structure of the portions of the TCB which are outside the reference monitor; in LOCK terminology these are called "Kernel Extensions (KEs)." In particular, Type Enforcement enables KEs to be implemented and certified as independently evaluated subsystems. Such evaluation can be done with confidence that such subsystems will be installed in a context which enforces the structural properties upon which their evaluation depends.

The ability to treat KEs as independently evaluated subsystems then makes LOCK an open architecture with regard to both implementation and assurance: new subsystems can be developed, evaluated, and added to a LOCK TCB in an incremental fashion.

1.3.3 Stronger Mathematical Foundations—The LOCK project, from its very beginnings as an implementation study for the Provably Secure Operating System in 1979, recognized the need to advance the state of the art of mathematical foundations for computer security. The goal of a SIDEARM module whose assurance remains valid for several generations of central processors intensifies this need: a module which is intended to be deployed for decades cannot depend on foundations which are known at the time of its design to be grossly insufficient.

The LOCK approach toward foundations has been to use a newer concept called "non-interference" and accept the risk associated with developing new strategies and towards proofs and using verification environments. This approach has given the project deeper insights into the nature of computer security, and greater confidence that the design of the SIDEARM will stand the test of time. An essential part of achieving rigor of mathematical results is the submission of those results to the so-called "social process" of refereeing by interested and competent parties; the LOCK project intends to follow this traditional mathematical procedure and has established a team of independent referees.

2. ELEMENTS OF AN INFOSEC SYSTEM

The LOCK architecture then defines the following three elements of an Infosec system, which are roughly "layered" on each other:

2.1 Secure Hardware Platform

The bottom element is the hardware which is necessary (but not sufficient) to process securely. This hardware base consists of the elements in an insecure machine (CPU, bus, memory, peripherals) enhanced by a properly-integrated SIDEARM and BED. In addition, the platform must have running on it a small amount of "glue software" which insures that the low-level, security-relevant functions are performed properly. This software consists, in general, of code that performs physical resource handling (e.g., process multiplexing, paging) in a manner that unavoidably involves the SIDEARM. The placement of the BED follows current cryptographic doctrine, which requires a physical architecture that insures that the cryptography cannot be bypassed.

2.2 Independently Evaluated Subsystems (a.k.a. Kernel Extensions)

The independently evaluated subsystems execute on the host processor, are controlled and protected by the SIDEARM, and are security-relevant for one or more of the following three reasons:

2.2.1 Required by Criteria—Certain subsystems are security-relevant because they satisfy a specific requirement which is called out in the relevant criteria for computer security or cryptography. Examples of such subsystems are labelers, user authentication packages, and System Security Officer utilities.

The nature of the evaluation process is such that these subsystems will usually be evaluated as part of the integration of the SIDEARM and BED to produce a TCB, and therefore the term "independently evaluated" may seem a misnomer. There is no technical reason, however, why a LOCK system could not be evaluated in stages, with a rating given to the integration of the SIDEARM and BED in one stage and the other required, host-resident subsystems (KEs in the LOCK terminology) in separate stages. The evaluation of the assemblage as a TCB would then consist of a review of the ratings of the individual components and the final integration.

2.2.2 Implement a Refined Security Policy—Subsystems can be security-relevant because they refine the resource-oriented security policy called out in the TCSEC and enforced by the SIDEARM. Such subsystems relate to the reference monitor in much the way that a DBMS relates to an operating system. Both a DBMS and an operating system do the same kinds of things: manage storage and process resources, provide names whereby data can be accessed for either observation and update, and provide a set of extended operations for data manipulation. They differ in the granularity of the data being named and the sophistication of the operations. Operating systems provide an environment of relatively large objects, with relatively simple naming conventions and a small number of operations. DBMSs provide a richer set of ways to name and manipulate individual data items.

Likewise, a reference monitor enforces a policy over what is basically an operating system environment of processes and files. The subsystems in this class refine that policy to apply to smaller items and uses more refined rules. A good example of such a subsystem is a DBMS with inference and aggregation controls. Such a DBMS can be designed (with suitable isolation mechanisms, like Type Enforcement in LOCK) to act as a data classifier which makes an upgrade decision based on the combination of data items being requested. The subsystem does not violate the reference monitor policy (because it always upgrades or leaves things the same) but it has a more refined set of rules which takes into account the fact that two items of data taken together may convey information whose sensitivity exceeds that of the data taken separately.

2.2.3 Downgrade—A downgrade situation exists in LOCK whenever a subject is given write access to an object whose security level is dominated by that of the subject. LOCK does not permit the theoretical case of the object level being noncomparable to that of the subject; accesses in LOCK must be "along" the lattice of security levels and cannot "jump across" to arbitrary levels. This restriction is imposed in support of decentralized administration of the lattice.

A downgrade situation is only permitted for those domains who have "trust" in the sense used by Bell and La Padula: their programs have been verified to exercise this potentially unsafe combination of accesses in a safe fashion. A downgrade situation may be safe for one or more of the following reasons: the amount of data which is downgraded is minimal, the act of downgrading is audited, the data which is downgraded is filtered to minimize the potential for use as covert channel, or the act of downgrading unavoidably involves human review. Downgrading typically occurs as a side effect of traditional operating systems functions, such as returning completion status for requests.

2.3 Applications

The term "applications" refers to all the software which executes in user mode and which has not been evaluated. Such software will have its accesses to objects controlled by the SIDEARM and may require services from independently evaluated subsystems (KEs). Applications software falls into three broad classes:

2.3.1 Encapsulated Single-Level Applications—This class of software is not "aware" of the fact that it is executing on a TCB rather than an ordinary system. In particular, it ignores the labelling of objects with security levels, which is the most important distinction between a TCB and an ordinary machine. Most of the software in this class is pre-existing, "ported" applications and operating systems. The LOCK TCB encapsulates and isolates multiple logical instances of such software at multiple security levels, and any sharing of data between different levels must be done by utilities which are separate from the application.

The environment provided for such applications resembles that of a "virtual machine monitor," which allocates resources to independent virtual computers, each of which runs a logically distinct copy of the operating system. The difference is that the separation is done on the basis of security level and enforced by the SIDEARM. The initial Unix capability for the LOCK prototype [3] will be done in this fashion.

2.3.2 Label-Recognizing, Single-Level Applications—This second class of applications is distinct from the first in that the software has been designed or modified to take into account the security labels on data. By the definition of applications code, such programs are unevaluated; there exist therefore strict limits on the kinds of label-based processing such software could perform. No such applications are planned for the LOCK prototype, but the class is consistent with the definition of a TCB and has been included in the set for logical completeness.

2.3.3 Multilevel Applications—Applications in this class will resemble a TCB in the sense that they will be split into evaluated and unevaluated subsets. They differ from a traditional TCB in that the two subsets act in concert to perform a specific applications function (e.g., intelligence data fusion and display) rather than acting as a general-purpose, resource-managing operating system. The evaluated subset, which comprises an independently evaluated subsystem or Kernel Extension, will be security-relevant for one or more of the reasons given in Section 2.2. A multilevel DBMS with inference and aggregation controls has been designed in this fashion and its implementation is planned for the LOCK prototype [4].

2.4 Assurance Considerations

Each of the three elements of the LOCK technology has its own assurance requirements:

2.4.1 Hardware Modules—The SIDEARM and BED can be viewed from an assurance perspective as a set of pre-assured functionality which cannot be tampered with. This, of course, is the most attractive thing about them: it relieves the vendor who wishes to incorporate them in a given machine from having to become expert in all the nuances and interpretations of the various criteria.

2.4.2 Independently Evaluated Subsystems—The assurance steps which would be applied to this software is similar to that which is applied to the SIDEARM. A security policy is formulated in terms of a formal model of processing and an argument submitted that the policy properly reflects the real-world policy in the area of interest. The subsystem is formally specified and an argument submitted that the specification exhibits the properties defined by the policy. A second set of arguments is submitted after implementation to show that the implementation corresponds to the specification.

The assurance steps differ primarily in the nature of the policy: the SIDEARM policy is very general and deals with access to representations of information; policies for independently evaluated subsystems will be specific to the service provided by that subsystem. In addition, the structure of the subsystem must be specified in a manner consistent with the Type Enforcement policy.

The factoring of the proofs of subsystems, as described in Section 1.3.2, is intended to simplify the assurance process by enabling subsystems to be demonstrated safe on the basis of significantly weaker properties of the code itself, because of the relatively strong statements that can be made about the context in which the software will execute. [5]

2.4.3 Applications—An "application" is not security-relevant and therefore its assurance steps are outside the scope of this discussion. It is worth noting that essential applications may undergo an assurance process similar to that of independently evaluated subsystems, and it is possible that LOCK configurations will be deployed which will run at a single security level but will make use of the Type Enforcement mechanism for high-assurance applications. Such configurations are likely for applications conforming to the general class of integrity policies described by Clark and Wilson [6] [7].

2.5 Vendor Characteristics

As discussed above, the LOCK approach permits each element of the technology to be offered by separate vendors:

2.5.1 Platform Vendors—LOCK hardware platforms would be attractive products for the traditional "hardware houses," which describes both independent companies or divisions of computer vendors. These organizations would assemble chips, boards, devices along with the LOCK hardware modules and the "glue software" described in Section 2.3 into a "bare-bones" machine.

2.5.2 Evaluated Subsystems Vendors—Evaluated subsystems fit more with the product capabilities of specialized software activities. Such activities exist either as free-standing companies or operations in a larger entity. Evaluated subsystem vendors would differ from ordinary software developers in having specialized in security concerns. They would be intimately familiar with the various criteria, regard their experience in having been evaluated as an organizational asset, and have a strong capability in the use of formal methods and other NCSC-approved assurance techniques.

2.5.3 Applications Vendors—These vendors could run the gamut from small software houses to large systems integrators; their unifying characteristic would be an unfamiliarity with (and in general disinterest in) security criteria and evaluation procedures.

3. INFOSEC INHIBITING FACTORS

If the term "independently evaluated subsystem" were to be replaced with "operating system" and "secure/security" replaced with "hardware details," then the three-tier vendor structure described above strongly resembles that which led to the explosive growth of the PC industry, where the availability of affordable hardware platforms and operating systems spawned a growth in applications software. This in turn increased the attractiveness of the platforms, which increased the number of platform (clone) vendors, decreased cost, and increased the potential market and therefore the number of applications. The result was a positive feedback loop which has not exhausted its energy to this day. Such explosive growth has not happened in the Infosec arena, and signs of it are not visible on the horizon; it is accordingly interesting to examine why.

3.1 Inhibitions for Platform Vendors

3.1.1 Short Product Cycles—The essence of most platform vendor's competitive position is the ability to respond rapidly to marketplace pressures and technological advance. A development cycle which includes independent evaluation is viewed by such vendors as being controlled by an outside organization, and having the risk that entire engineering investment can be lost because someone else made them late to market.

It is worth noting in this regard that implied in the notion of an Evaluated Product List has been the promise that the secure versions of a given system would be technologically and price-performance comparable to the insecure ones; a promise which has never been made in the cryptographic area, where the effects of freezing a technology for evaluation purposes has long been accepted as an unavoidable price of security. The implied promise of "secure and competitive" was much more reasonable when the computer market was dominated by the large mainframe vendors because they leased equipment to their customers and their profitability increased with the length of time a given piece of equipment stayed in service. The implied promise is much more questionable in a multi-vendor, purchase environment where profitability increases with turnover in equipment and consequent short product cycles.

3.1.2 Cryptophobia—Many potential platform vendors are wary of becoming involved with the development of an endorsed cryptographic capability for their machines. They do not have the facilities to protect classified information or the ability to staff projects exclusively with U.S. citizens. They are put off by the inability to forecast the time and effort required to pass evaluation, and the perceived rigidity of cryptographic technology deprives them of their accustomed freedom in making engineering tradeoffs. They also are reluctant to invest heavily in products for which there is a limited export market.

3.1.3 Need for Evaluated Subsystems—There currently is no precedent for evaluating a hardware base as a component. Any vendor seeking a rating must implement and verify evaluated subsystems in order to satisfy the totality of the TCSEC requirements. Development and formal verification of software is an activity which is completely alien to the culture, management, design automation facilities, financial structure and risk management practices of the typical small, specialized hardware vendor. The vendor is then forced to seek a subcontractor or teaming partner, which is an equally alien practice to many vendors.

3.2 Inhibitions for Evaluated Subsystem Builders

3.2.1 Lack of Accessible Platform Technology—An independently evaluated subsystem can only be implemented and evaluated in the context of a secure platform which contains at least a reference monitor. The small number of existing reference monitors have enforced security policies which are, at best, marginally adequate to support the independent implementation and evaluation of security-relevant software. In addition, the current concept of the Evaluated Product List, with its emphasis on vendor investment, means that the technical details of the reference monitors remain proprietary. Thus every TCB which appears on the Evaluated Product List becomes a "closed" architecture in the legal if not the technical sense.

3.2.2 Narrow, Platform-Specific Market—Even if an arrangement is made between a potential vendor of an independently evaluated subsystem and a TCB developer, the market potential of that subsystem is limited to the installed base of that specific TCB, which is a subset of a subset of the market for other software. The lack of formal or *de facto* standards for reference monitors strongly inhibits the portability of independently evaluated subsystems, which in turn reduces the incentives of software organizations to acquire the special verification expertise and patience with the evaluation process required to develop them.

3.3 Inhibitions for Applications Builders

The "applications" referred to in this section are those described in Sections 2.3.2 and 2.3.3 above, that is, those which take the labelling of data into account in their processing. The inhibitions to their development duplicate exactly those described for evaluated subsystems, with the difference that it is the characteristics of the entire TCB and not just the reference monitor subset which is the issue.

4. LOCK COUNTERS TO INHIBITING FACTORS

The overall goal of the LOCK project is to "prime the pump" for the kind of deployment of technology into the user community, recognizing the current computing environment of specialized vendors and short product cycles. The LOCK effort is taking the following steps to counter the inhibiting factors described in Section 3:

4.1 Independent Subsystems, Independent Vendors

The LOCK security architecture is designed to be open and highly modular, and enforces a security policy (Type Enforcement) which supports the integration of independently evaluated subsystems. The LOCK technology, then, is intended to be a TCB-builder's tool rather than a single instance of a TCB.

A potential TCB builder should be able to take a "mix and match" attitude toward the various LOCK elements. Lower-rated TCBs could incorporate the SIDEARM but not the BED; other vendors, wishing to implement components to be evaluated under the Trusted Network Interpretation, might incorporate the BED but implement a more traditional reference monitor in software. Their options range from a single package to provide a network component function, through special-purpose TCBs, to "full-up" TCBs which provide all of the services of a contemporary operating system. These options can be exercised either by a single developer/vendor, or by a systems integrator who purchases a hardware platform from one vendor, independently evaluated subsystems from a variety of other vendors, and provides the traditional value-added service of integration and interface development.

4.2 Standards

The options described in Section 4.1 exist because the LOCK technology includes both formal and *de facto* standards. The formal standards will be the Interface Control Documents for the SIDEARM and the BED. The *de facto* standards will be the various structuring approaches toward independently evaluated subsystems which arise from the use of the Type Enforcement policy. Standards permit the independence of vendors of the various elements, which is a prerequisite for the kind of positive feedback loop in the marketplace that LOCK intends to initiate.

4.3 Cost and Risk Reduction for Potential Vendors

A technical potential for vendors to provide elements of secure systems will remain just a potential until business decisions are made. A significant, if not overwhelming, factor in such decisions is the risk associated with an engineering investment. High risk is a far more inhibiting factor than high cost: if the cost is high, and confidence in the cost figure is equally high, then it is possible that some vendor, of some size, will find a market whose return will justify the cost. If the risk is high, then the potential cost becomes unbounded, and prudent organizations will seek more stable markets. The LOCK project is aware of this situation and has incorporated the following attributes which reduce the perceived and actual risk assumed by potential vendors of platforms, independently evaluated subsystems, or multilevel applications.

4.3.1 Useful Technology Transfer to Platform Vendors—Platform vendors are essential because it is the availability of platforms, like the availability of "bare-bones" PCs, which initiates the technology infusion. The most important attribute of LOCK from the point of view of these vendors is the amount of technology that is transferred to them and the form in which it comes.

The physical modules which comprise the SIDEARM and the BED encapsulate a large number of design decisions and issues, some highly classified; no knowledge of them is required to be a platform vendor. The vendor's concern is in interfacing these modules properly (i.e., unavoidably) into the data paths of a given architecture. How the modules go about their functions, and more importantly why, is not a concern of the vendor.

A different situation exists for vendors attempting the development of a higher-rated TCB for the current Evaluated Product List. Such vendors must start essentially from scratch, and develop an in-house or subcontract/team capability for all aspects of policy definition, assurance, and coping with the evaluation process. Such development requires a significantly greater technical capability, management skills, and capital equipment than that which such vendors are typically willing to devote to a market which does not yet exist.

The ability for a vendor to produce a secure variant of a machine without expanding organizational capabilities is essential if the smaller hardware houses are to be encouraged to become secure systems vendors. Such organizations are attractive candidates as vendors because their smaller size gives them a lower overhead and makes them more flexible and responsive to market forces. They are also likely candidates in that they will be motivated by a desire to carve out distinctive market niches. The encapsulation of security expertise in standard hardware modules will enable such vendors to enter the security marketplace and still maintain the austere engineering establishment which makes them successful.

4.3.2 Worked Examples—The LOCK project provides potential vendors with worked examples of documentation, and experience in design, implementation, and evaluation. These examples are not available through the traditional Evaluated Product List process because they are proprietary to the vendors who have submitted their products for evaluation.

In the documentation area, the LOCK project has already produced and made publicly available the so-called "Annotated TCSEC," which is a set of generic interpretations of the TCSEC that potential vendors, particularly those new to the security field, can study to gain insight into the evaluation process. The project has developed an innovative format for the DTLs and FTLs which will be available, without proprietary restrictions, to potential vendors. The verification results and covert channel analysis will also be available to appropriate parties.

Of equal importance will be the worked examples of the documentation submitted for cryptographic evaluation. A Theory of Equipment/System Operation has been completed and the companion Theory of Compliance will also be delivered to the NCSC with unlimited data rights. These latter documents will assist future vendors not only in the implementation of a LOCK platform but will also facilitate the development of other products under the Commercial Comsec Endorsement Program.

The LOCK prototype will provide a worked example of how to integrate SIDEARM and BED modules into a typical architecture. A particularly important role of the prototype will be in locating performance bottlenecks, an activity which history has proven can only take place on an actual machine under actual loads. The project will also demonstrate how independently evaluated subsystems are implemented and verified, and provide worked examples of how the security-relevant subset of a multilevel application is determined and interfaced to the rest of the application.

4.3.3 Reuse of Technology—The LOCK technology has the promise of progressively reducing the cost and schedule required to produce secure machines, owing to the availability of pre-evaluated, encapsulated modules which are integrated into several generations of a variety of architectures. In the current situation, each TCB developer starts from scratch, and the combination of machine-dependent software solutions and proprietary restrictions effectively prevents any "learning curve" phenomenon from taking place.

It is likely that the evaluation process will cause secure variants of any machine to always lag behind the unsecured machine to the marketplace. The availability of preexisting technology in module form raises the possibility that secure variants of machines can be considered early in product planning cycles and thereby reduce the lag.

Finally, the existence of the formal and *de facto* standards will permit the porting, at some expense less than complete implementation and evaluation, of independently evaluated subsystems from platform to platform. Thus each succeeding platform can be more attractive to potential customers because it can have much of its predecessor's software available as well as the new subsystems which distinguish it in the marketplace. Similarly, the production of each new platform is progressively more attractive to potential vendors because of the ability to capture an upgrade or expansion market of existing systems.

4.3.4 Useable Platforms for Developers—Successful software development is almost always the result of small teams, and the way to build an inventory of software quickly is to encourage a large number of such teams to work on separate elements in parallel. This is, of course, what took place in the PC arena, in which a multitude of individual vendors sprung up almost overnight.

Two things are required for such a phenomenon: the existence of formal and *de facto* standards to allow reuse of software and porting, and the availability of platforms for use in the latter stages of implementation, test, and integration. The LOCK technology provides the former and the LOCK prototypes provide the latter. With prototypes available to software developers, there exists the strong possibility that an inventory of applications and independently evaluated subsystems can be offered by a variety of vendors simultaneously with or soon after the introduction of product-quality LOCK platforms.

5. TECHNICAL DEMONSTRATIONS AND CONSTITUENCIES

Each of the above classes of vendors, along with the evaluation community and the LOCK program management team, is a constituency to which different aspects of the LOCK technical demonstrations have different degrees of importance.

5.1 Technical Demonstrations

The LOCK plan contains two technical demonstrations, scheduled at the 24 month and 42 month points in the project, and referred to as the "24-month demo" and the "42-month demo." The 24-month demo date is 1 March 1989 and the 42-month demo date is 1 October 1990.

Technical demonstrations were included in the program plan to provide a "forcing function" for technical progress and a means whereby that progress could be evaluated. Since they represent points at which significant attention will be directed toward the project as a whole, they are not events which can be separated from the demonstration goals of the project as a whole. It is necessary, therefore, to examine the range of constituencies to which LOCK is demonstrating results, and consider what is to be demonstrated to each constituency during the two "windows of attention" which the technical demonstrations will open.

Demonstrations will consist of the delivery or publication of results as well as the actual "demo event" witnessed by management; published results are accordingly part of the technical demonstrations.

5.2 Constituencies

The constituencies to which the LOCK program is demonstrating results consists of the three classes of potential vendors described in Section 2.5, as well as the general evaluation community and the LOCK project management team. The evaluation community is defined as all outside technical observers of the project, including the research community, accrediting authorities, and those responsible for the formal evaluation of the prototype. The LOCK project management team includes both contractor and customer management and any other parties concerned with the cost, schedule, and technical performance of the project.

6. 24-MONTH DEMONSTRATION

6.1 Objectives By Constituency

6.1.1 Platform Builders—At 24 months the project intends to demonstrate to potential platform vendors that it is possible to integrate a SIDEARM module into a machine in the manner dictated by the LOCK architectural concepts. In addition, there is an objective to show that such a hardware reference monitor can be generic to a range of processor and bus architectures. Finally, the project intends to show that a BED which performs media encryption in a controller-independent fashion can be incorporated into an existing machine.

The feasibility of a SIDEARM will be demonstrated by exercising a prototype containing such a module. The generic nature of the SIDEARM design will be shown by submitting the SIDEARM Interface Control Document to review by potential vendors. The feasibility of a BED design will be shown by presenting a variety of design documentation, as well as displaying implemented but not yet integrated BED prototype hardware, to the technical community.

6.1.2 Evaluated Subsystem Builders—The LOCK project intends to show potential vendors that independently evaluated subsystems are a technically feasible way to enter the secure software marketplace. Central to this demonstration is the reasonableness of the Type Enforcement policy and its notion of an "assured pipeline" as a software structuring approach. The principal vehicle for this demonstration at 24 months will be the availability of a variety of design documents for independently evaluated subsystems structured using the assured pipeline method. These will include the subsystems to be incorporated in to the LOCK TCB, such as labelers and SSO utilities, as well as the multilevel DBMS.

6.1.3 Applications Builders—The first set of applications to be available on LOCK will most likely be single-level subsystems of the kind described in Section 2.3.1. The LOCK project intends to attract the attention of the vendors of such applications and show that LOCK provides a congenial environment for their software. The principal vehicle for this demonstration will be the effort to port an encapsulated single-level Unix onto LOCK. At 24 months, the project will provide design documentation, and reports on the progress of the effort by the porting subcontractor.

6.1.4 Evaluators—The LOCK project has two major results to show to the formal evaluation community. The first of these is that LOCK is a "proper superset" of the A1 criteria, that is, that it meets and exceeds all requirements. The second is that an Infosec evaluation is feasible; this latter result requires that the interface between the computer security and cryptographic evaluation criteria be rationalized for the special case of the BED and its integration into the LOCK prototype.

Compliance with the TCSEC will be demonstrated at 24 months by an Annotated TCSEC whose interpretations are acceptable to the evaluators and report on progress and the status of outstanding issues in the developmental evaluation. The feasibility of an Infosec evaluation will be shown at 24 months by the deliver and acceptance of the Theory of Equipment/System Operation for the BED, which will define the allocation of requirements from the computer security and cryptographic criteria.

The principal result to be shown to the research and informal community of interest in LOCK is that of the rigor of the mathematical analysis. This result will be demonstrated at 24 months by the establishment of the refereeing team and a progress report on their initial reactions to the analysis.

6.1.5 Project Management—The obligation of the LOCK project to its own management is to show tangible results. At 24 months, the scheduled tangible results are that the SIDEARM has been integrated into the host machine and is enabled. This goal has been stated informally as showing that "the system breathes and breathes securely," where "securely" is taken to mean that the computer security mechanisms are in place and are unavoidably encountered by software running on the host. This requires a demonstration that the integration of the SIDEARM and the "glue software" on the prototype has successfully passed the critical point where all elements are present and working together, albeit awkwardly and with suboptimum performance.

The demonstration will be conducted in two parts, a functional demonstration and a security demonstration. The functional demonstration will consist of a simple, hands-on exercise of the system. The security demonstration will show that the action's subverted software are intercepted by the LOCK security mechanisms.

It should be emphasized that the 24-month demo is a laboratory demonstration of a prototype and is in no sense a product rollout. In particular, there will be no attempts to show that the performance of the LOCK system is relative to an unmodified machine at this time. There are several reasons for this omission.

1. The system will be extremely immature and there will have been no opportunity for any optimization.
2. The availability of the Type Enforcement facility has led programmers to (properly) apply the concept of "least privilege" to the software structure, which in turn increases the amount of processor context switching in a given task.
3. The performance impact of closing covert channels cannot be known until an operational system is instrumented and run under varying loads.

Given all these uncertainties, it was decided not to schedule a demonstration of performance, with the implication of a set performance goal, at the 24 month milestone.

6.2 Demonstration Scenario

6.2.1 Functional Demonstration—The functional demonstration will consist of the minimum amount of functionality required to show that the essential elements of the system have been integrated with each other. The demonstration will use "throwaway" software which will interface directly to the reference monitor. There will be no Kernel Extensions available and the configuration of subjects will be "hardwired", so that no logon, either actual or simulated, will be performed.

The functional demonstration will show that the system is capable of entering, storing, and displaying data. To this end a very minimal text editor will be programmed in the Small C language used for the security demonstration. The text editor will have minimal editing commands and enter its data into a single, dedicated segment. This latter restriction relieves the editor from having to recognize any file names or do any explicit movement of data from buffers to permanent storage.

When the editor starts up it will paint the screen with the current contents of the dedicated "save segment." At initial startup this segment will be empty and the screen will be blank except for labels at the top and bottom which denote the security level of the information being displayed. The functional demonstration will consist of bringing the system up, entering data, taking the system down, and observing on bringing it back up that the data has been saved. The demonstration will be accompanied by a presentation which shows the elements of the system invoked by the demonstration software.

6.2.2 Security Demonstration—Security demonstrations are generally unspectacular because security deals with events that do not happen. The LOCK security demonstration will attempt to alleviate this shortcoming by being threat-based and using a more elaborate attack scenario.

The scope of the demonstration at 24 months will be the Mandatory Access Control and Type Enforcement facilities. Discretionary Access Control will not be demonstrated because the necessary Kernel Extensions will not have been implemented.

The demonstration will be piggybacked on the functional demonstration, and will use two subverted versions of the text editor used in the functional demonstration. The Mandatory Access Control demonstration will show that one subverted editor can steal keystrokes and display them to an unauthorized person on an unmodified machine but is intercepted on the LOCK prototype. The Type Enforcement demonstration will show that a second subverted text editor can tamper with the labelling on the screen of an unmodified machine but is prevented from doing so on the LOCK prototype.

The elaboration consists of the manner in which the subverted code is inserted into the text editor. For the narrow purposes of the demonstration, it would be sufficient to hand-code the subversion routine into the source; in the interests of a more graphic demonstration, a subverted Small C compiler which implements a "Thompson virus [8]" will be used. The demonstration will then consist of compiling an apparently clean compiler from source code, then using that to compile the editor, and then showing that the attacks mounted by the compromised editors succeed on an unmodified machine and fail on the LOCK prototype.

7. ACKNOWLEDGEMENTS

Work on LOCK has been supported in whole or in part by U.S. Government contracts MDA904-80-C-0470, MDA904-82-C-0444, MDA904-84-C-6011, MDA904-85-G-2035, and MDA904-87-C-6011.

8. REFERENCES

1. Saydjari, O.S., Beckman, J.M., and Leaman, J.R., "Locking Computers Securely," Proc. 10th Nat. Comp. Sec. Conf., NBS, 1987, pp. 129-141.
2. Boebert, W.E. and Kain, R.Y., "A Practical Alternative to Hierarchical Integrity Policies," Proc. 8th Nat. Comp. Sec. Conf., NBS, 1985, pp. 18-27.
3. Schaffer, M.A., and Walsh, G., "LOCK/ix: On Implementing Unix on the LOCK TCB," these proceedings.
4. Stachour, P., Thuraisingham, B., and Dwyer, P., "Update Processing in LDV: A Secure Database System," these proceedings.

5. Young, W.D., Telega, P.A., Boebert, W.E., and Kain, R.Y., "A Verified Labeler for the Secure Ada Target," Proc. 9th Nat. Comp. Sec. Conf., NBS, 1986, pp. 55-61.
6. Clark, D.D., and Wilson, D.R., "A Comparison of Commercial and Military Computer Security Policies," Proc. Symp. on Sec. and Priv., IEEE, 1987, pp. 184-194.
7. Report of the Workshop on Integrity Policies for Computer Information Systems, to be published by NBS.
8. Thompson, K., "Reflections on Trusting Trust," Comm. ACM 27,8, Aug. 1984, pp. 761-763.

Constructing an Infosec System Using LOCK Technology

W. E. Boebert

Honeywell
Secure Computing Technology Center
2855 Anthony Lane S.
Minneapolis MN 55418

Abstract

In this paper we describe the elements of the LOCK technology and discuss how they are embedded in a host computer system to produce a compatible Infosec version of the machine.

1. BACKGROUND AND OVERVIEW

The overall goal of the LOCK project is to provide a technology for the development of Infosec products. The technology is intended to be processor-independent and include integrated cryptography. Its architectural approach separates the traditional TCB into a hardware reference monitor, which mediates access to system resources such as segments and devices, and software "Kernel Extensions," which are constrained by the reference monitor and which perform applications-dependent, security-relevant functions such as the labelling of output. Kernel Extensions are protected by an extension to the traditional Bell and La Padula security policy called "Type Enforcement," in which the hardware reference monitor enforces a defined data flow structure on the software. An overview of the LOCK project and a description of the type enforcement policy is given in Reference [1].

If successful, the LOCK architecture should establish a de facto standard which will benefit both the vendors and the users of Infosec products. The separation between the hardware platform and the software Kernel Extensions will permit hardware and software vendors to concentrate on what each do best. Customers will have the freedom to choose from a variety of platforms, enabling them to select Infosec products which are processor-compatible with their other systems. Customers such as systems integrators will be able to choose platforms from one vendor, Kernel Extensions from a second, and write both applications and security-relevant code themselves. They will need submit for evaluation only those extensions which are new, enabling Infosec systems to be deployed to end users at a fraction of the budget and schedule required to develop them from scratch. The spread of the LOCK architecture and concepts across a wide range of vendors will encourage customers to select Infosec solutions, which in turn will provide positive feedback to vendors by increasing the potential market.

The LOCK project is currently in a proof-of-principle phase which consists of constructing an Infosec prototype using a Motorola 68020 processor and a VME bus. An important aspect of the proof of principle is the demonstration that it is feasible to apply LOCK technology to other processors; this paper, then, is both a guide and an invitation to potential vendors of Infosec products.

2. THE LOCK ARCHITECTURE

2.1 Basic Partitioning

As mentioned above, a LOCK architecture is split at the top level between a reference monitor, which "owns" all the basic resources of the system (e.g., segments, devices, processor cycles) and Kernel Extensions, which perform security-relevant functions such as output labelling. Access by Kernel Extensions to resources are mediated by the reference monitor. In addition to the reference monitor and Kernel Extensions, there is a general class of software called "applications" which perform no security relevant function and which are presumed by the security policy to contain hostile programs. The term "applications" is used simply to distinguish the unverified from the verified portion of the system, and should not be taken to imply the traditional meaning of "end user programs." For example, the Unix operating system will run as an "application" on top of the prototype LOCK TCB.

2.2 Access Mediation and Cryptography

The reference monitor utilizes both the traditional Compusec technology of access mediation and the traditional Comsec technology of cryptography to achieve an Infosec result. Roughly speaking, access mediation is used to protect data in "red" (intelligible) form (which it must be in in order to be processed) from improper access by potentially hostile code. This protection follows the doctrine defined in the Bell and La Padula model and called out in the TCSEC. Cryptography is used to store data in "black" (unintelligible) form for those periods of time when no program needs to access it, e.g., when it is "swapped out" to disk. Storing data in black form on magnetic media reduces the physical security requirements on the system, for the data can quickly be rendered unusable by the destruction of the applicable keying material.

In addition, cryptography is used to close covert channels, protect security-critical data bases, and defend against attacks by subverted device controller hardware and firmware. Both the access mediation and the cryptographic subsystems of the reference monitor are isolated in distinct hardware to provide physical and electronic protection and to facilitate verification. This requirement to isolate results in the generic LOCK architecture shown in Figure 1.

2.3 Access Mediation Subsystem

The access mediation subsystem consists, in the prototype, of a commodity Memory Management Unit (MMU) and a coprocessor called the SIDEARM. The SIDEARM contains the security data bases and logic used to decide what access a program in execution (subject) shall be granted to a system resource (object). This decision is manifest as a value in the access control field of the appropriate entry in the table the MMU uses to convert "virtual" program addresses into "real" bus addresses.

A mechanism, whose characteristics depend on the processor and MMU design, must therefore be provided to link the SIDEARM with the MMU. This mechanism must insure that the SIDEARM is consulted whenever an MMU entry is to be made and that the values loaded in the MMU accurately reflect the decisions of the SIDEARM. Generally, this mechanism will take the form of small amount of privileged and verified "glue code" which runs in "master mode" or "ring 0" on the processor.

Objects are identified within the SIDEARM by unique identifiers, or UIDs. The SIDEARM accordingly views low-level memory as a "flat" file system of objects denoted

by UIDs, and is most compatible with virtual memory schemes in which a uniform address space is "seen" by the processor.

2.4 Cryptographic Subsystem

The cryptographic subsystem is called the Bulk Encryption Device, or BED. It utilizes endorsed cryptography and is designed to conform to current cryptographic doctrine. This doctrine dictates a physical separation of red and black data by the cryptographic device. The BED is accordingly positioned between a red and a black bus, with the unverified controllers segregated on the black bus. In addition to the cryptographic transformation, the BED performs special operations to "seal" data to its physical address on the black bus. Data is therefore protected both from observation by subverted controllers and from attacks in which classified data is moved clandestinely into unclassified segments. A second cryptographic device, called the SIDEARM Encryption Unit, or SED, is used to protect the security data base and audit media from unauthorized observation. The detailed placement of these cryptographic subsystems is shown in Figure 2.

2.5 Generic Architecture

Both the SIDEARM and BED are designed to be as generic as possible. The SIDEARM interfaces with a bus-dependent module called the Host Interface Processor by means of a generic 64 bit wide bus and FIFO buffers. The BED has distinct bus-dependent interface areas and internal logic to make it appear as an I/O device to processors and as a processor to I/O devices.

3. IMPLEMENTING A LOCK SYSTEM

3.1 Requirements on the Base Machine

In order for a computer to be a candidate base for a LOCK system, it must have the following characteristics:

1. It must provide a means for mediating accesses by the processor to memory. At a minimum the machine must have the ability to enforce distinct read and read/write access modes. Such enforcement is provided by most contemporary MMUs as a side effect of address mapping.
2. It must have an "open" interface for I/O devices, so that the SIDEARM coprocessor can be attached. Such a facility is provided by most contemporary bus-oriented architectures.
3. There must be some way to establish a verified path between the SIDEARM and the mediating mechanism (e.g., the MMU), so that accesses are mediated only in those ways dictated by the SIDEARM. This path can take the form of a direct, "back door" interface into the MMU's data structures, or a more circuitous route using the "glue code" approach described above.
4. It must provide a means for isolating a small subset of software and associated data objects from tampering or accidental change. The most common isolation mechanisms involve "multistate" processors which have a distinguished "master mode" or "ring 0" state. This state is typically entered by a trap mechanism and the processor must be in this state in order to perform privileged operations and access kernel objects. It is extremely desirable that the host architecture permit "master mode" code to be stored in ROM instead of RAM.

5. Its protocol for transfer of data between peripherals and memory must permit the interposing of encryption as an intermediate step. Most contemporary bus protocols, which are designed to accommodate a wide range of transfer rates and timings, can satisfy this requirement.

3.2 Conversion of a Base Machine to a LOCK Platform

3.2.1 Required Steps--The process of converting a base machine to a LOCK platform requires the following steps: First, the SIDEARM must be attached electrically to the machine. Then, the peripherals must be isolated on a separate black bus and the BED interposed between it and the red bus. Next, the communication between the SIDEARM and the MMU must be established. Finally, the code which allocates physical resources such as processor cycles to applications programs must be modified to close low-level covert channels. The result will be a transformation such as that shown in Figures 3 through 7. There will also be requirements for tamper-resistant and low-emanations packaging, which are outside the scope of this document.

3.2.2 SIDEARM Interface--The early versions of the SIDEARM will occupy a separate cabinet connected by cable to the host machine. The Host Interface Controller (HIC) which drives the cable must be able to arbitrate for the host bus and transfer data into the host address space. It must also be able to provide utility functions such as asserting interrupts on the processor bus, holding the host bus in reset or standby state, indicating impending power failure, observing self-test failure of host boards and sensing that the host processor is down. Design and implementation data for the HIC used on the prototype, which connects a SIDEARM to a VME bus, will be available to potential LOCK implementors.

3.2.3 BED Interface--The processor/memory (red) interface portion of the BED card must be able to respond to any host bus cycle that normally would be responded to by an I/O adapter. It must simulate all of the control registers of the I/O adapters and be able to analyze instructions for I/O transfers. In the case of memory to device transfers, it must be able to transfer the referenced data to its onboard memory for encryption. In the case of device to memory transfers it must be able to initiate and wait for decryption, transfer data from its on-board memory to main memory, and signal the processor that the I/O operation is complete.

The device (black) interface portion of the BED card must operate symmetrically to the processor/memory portion. It must be able to program the real I/O controllers to transfer encrypted data to and from devices and be able to recognize when such transfers are completed.

In addition to the customizing of the BED to a specific bus, device drivers must be written or modified to accommodate the BED logic. If the interfacing to the bus is done with proper respect for programming standards, this task should be no more difficult than that of supporting a new device controller for the unmodified machine. In general, the BED interface will be a superset of the command set for a non-cryptographic device, as it will have special commands for cryptographic control functions.

Finally, certain physical requirements for operator access to the cryptographic subsystem will have to be satisfied. This can generally be done by adding a panel about 3" by 6" in size for the necessary indicators, alarms, and human interfaces and cabling it directly to the BED. Design, implementation, and packaging data for the BED used on the the prototype will be available to potential LOCK implementors, as will the documentation and source code for the I/O drivers.

3.2.4 SIDEARM/MMU Interface--As mentioned before, this interface can be implemented as a hardware "back door" to the area of memory in which the MMU tables are stored or by means of software which runs on the host processor. As the latter will be by far the most common case (especially in light of the trend to integrated processor/MMU sets), it is the one that will be treated here.

This software must perform two basic functions: it must manage the interaction between the SIDEARM and the MMU and it must coordinate that interaction with the physical memory management, such as paging or segmentation. The nature of the interactions can best be described by example.

The most characteristic reference monitor request is "open object," in which a program in a known security context (subject) asks to have a data segment (object) added to its address space. The object to be added is denoted to this software by its UID. The interface software must then perform the following functions:

1. It must pass the UID and the request code across the HIC to the SIDEARM. At this point, for efficiency reasons, it will typically relinquish the host processor, i.e., dispatch some other process and wait for the HIC to signal completion of the request by the SIDEARM.
2. Upon resumption, (e.g., upon receiving the completion interrupt from the HIC) it must accept the allowed access mode returned by the SIDEARM, convert this into a form comprehensible to the MMU, and initialize the proper MMU tables accordingly. The SIDEARM will return a generic set of access modes on the assumption that it is dealing with an "ideal" MMU; the actual set for a specific machine will generally be smaller than this and hence a transform is required.
3. The interface software must also communicate with the memory management software to insure that the object is addressable. This will require steps such as initialization of page tables or bringing segments into memory from disk.

3.2.5 Physical Resource Management--In general, the physical resource management software will have to be replaced, either (as described above) to accommodate the use of the SIDEARM coprocessor or to close covert channels. The two areas most likely to be affected by the latter requirement are processor management and the interface to the internal clock.

In order to reduce the number of potential covert channels, the allocation of processor cycles to subjects (processes) must be as even as possible, so that there are few opportunities for one program to affect the timing of another. Dispatching algorithms which operate on the basis of fixed time slices are accordingly favored. Likewise, the resolution of the clock used by applications code must be coarsened substantially; the prototype reports time only to the nearest second. These changes, like any strategy used to reduce the number and bandwidth of covert channels, will reduce performance to some degree.

The experience of the prototype team is that the two sets of software changes described above (SIDEARM/MMU interfacing and covert channel closing) require a re-implementation of the entire bottom level of the software that runs on the machine. The experience of the prototype team, in the form of specifications, code, and reports, will be available to all potential LOCK implementors.

3.2.6 Kernel Extensions and Applications--Kernel Extensions are security-relevant software subsystems that run on top of the basic LOCK platform; applications are the rest

of the software required to do useful work with a LOCK system. In general, the distinction between the two is made depending on whether the software runs "encapsulated single-level" or "multilevel." This difference is best understood by considering the example of the Unix operating system.

Unix is treated as an application on LOCK prototype because it runs "encapsulated single-level," that is, an entire Unix environment is provided to a single user at a single security level. This approach maintains the maximum degree of compatibility with existing Unix software, because the programs do not need to "know" that they are running on a TCB. In particular, they do not need to make decisions based upon the security levels associated with the objects they access.

If, on the other hand, the LOCK version of Unix was modified to recognize security levels, then the code which implemented those modifications would (with rare exception) become Kernel Extensions and would have to be evaluated as free-standing subsystems before they could be installed in an operational environment. The resulting operating system would then be a true "multilevel Unix;" the simpler, "encapsulated single-level" approach was taken in the LOCK prototype in order to obtain a basic Unix capability in the shortest period of time.

(There exists a third option for the implementation of an operating system such as Unix, which consists of interfacing the lower levels of the operating system directly to the SIDEARM, MMU, and the BED. This option was not selected in the LOCK project because of the desire to maintain generality in the prototype; it may be a very attractive option for vendors whose sole interest is in an Infosec Unix.)

The LOCK prototype effort will develop and submit for evaluation a set of Kernel Extensions to perform standard TCB services such as security administration, login, and labelling of data. Full design and implementation data on these subsystems will be available to potential LOCK implementors. It is also likely that a large percentage of the code will transfer successfully to LOCK platforms using a different processor. In addition, the LOCK project will make available its methodology for the design, specification, and verification of Kernel Extensions.

In addition, a separate project called LOCK Data Views (LDV) is engaged in the design and implementation of a multilevel DBMS which incorporates controls on inference and aggregation. The LDV DBMS exhibits the typical LOCK partitioning between applications and Kernel Extensions; the results of the effort will be available to the technical community as an example of how higher-level Infosec systems are implemented using the LOCK technology.

4. SUMMARY

The LOCK project is developing a technology for the construction of Infosec systems. This technology consists of hardware modules, software modules, a worked example of integration into a M68020 platform, and worked examples of documentation and verification techniques. The technology is designed to be as generic as possible and not force either vendors or customers into a specific choice of processors. It is also designed to accommodate the incremental implementation and certification of extensions to a TCB. The purpose of the technology is to radically cheapen and shorten the process of producing Infosec systems, by reducing a previously arcane process to a straightforward one of integration of previously-developed elements into a working whole. It is this process which has been described here.

5. ACKNOWLEDGEMENTS

Work on LOCK has been supported in whole or in part by U.S. Government contracts MDA904-80-C-0470, MDA904-82-C-0444, MDA904-84-C-6011, MDA904-85-G-2035, and MDA904-87-C-6011.

6. REFERENCES

1. Saydjari, O. Sami, Beckman, Joseph M., and Leaman, Jeffrey R., "Locking Computers Securely," Proc. 10th Nat. Comp. Sec. Conf., 129-141.

Update Processing in
LDV: A Secure Database System

Paul Stachour, Bhavani Thuraisingham,
and Patricia Dwyer
Honeywell Inc., Minneapolis, Minnesota

November 29, 1988

Presented at

11th National Computer Security Conference,
Baltimore, Maryland, October, 1988

PREFACE

This paper gives an overview of the design of the Multilevel Secure Database Management System (MLS/DBMS), LOCK Data Views (LDV), for the Secure Distributed Data Views contract. The prime contractor is Honeywell's Secure Computing Technology Center (SCTC) and the subcontractor is Honeywell's Corporate Systems Development Division (CSDD).

This work was performed under contract F30602-86-C-0003 from the Distributed Systems Section/COTD of Rome Air Development Center. RADC's support of the LDV application, and their interactions with the National Computer Security Center LOCK program is gratefully acknowledged.

This paper summarizes activity performed by the LDV design team of Honeywell's Corporate Systems Development Division and Honeywell's Secure Computing Technology Center team on the Implementation Specifications task during the period 1 May 1986 through 1 April 1988. The Honeywell team included Patricia Dwyer, Emmanuel Onuegbe, and Bhavani Thuraisingham of CSDD and Paul Stachour and Tom Haigh of SCTC.

Table of Contents

1. Problem Statement	1
1.1. Problem: Database Security	1
1.2. Approach: Database Complementing OS	1
2. Security Policy Overview	1
2.1. Security Policy Requirements	1
2.2. LOCK TCB Security Policy	2
2.3. DBMS Security Policy Requirements	2
2.4. DBMS Policy Extensions	3
2.4.1. Update Classification Extension	3
2.4.2. Response Classification Extension	3
3. Pipeline Organization	3
4. Update Pipeline	4
4.1. Introduction	4
4.2. Design Issues	4
4.2.1. Data Distribution	4
4.2.2. LDV Data Distribution Scheme	6
4.2.3. Data Partitioning of Distributed Data	6
4.2.4. Assuring the Correctness of Responses In Spite of Polyinstantiation	7
4.3. Design Overview	7
4.3.1. Insert Request	7
4.3.2. Delete Request	12
4.3.3. Modify Request	12
4.3.4. Overview of the Major Modules	13
4.3.4.1. User Request Manager	13
4.3.4.2. Relational Access Manager	14
4.3.4.3. Execution Manager	14
4.3.5. Security Critical Modules	15
5. Conclusion	15
6. References	15

1. Problem Statement

1.1. Problem: Database Security

Within the Department of Defense (DoD), the number of computerized databases containing classified or otherwise sensitive data is increasing rapidly. Access to these databases must be restricted and controlled to limit the unauthorized disclosure, or malicious modification, of data contained in them. Present Database Management Systems (DBMSs) do not provide adequate mechanisms to support such control. Penetration studies have clearly shown that the mechanisms provided even by "security enhanced" database systems can be bypassed, often due to fundamental flaws in the systems which host the DBMS. This has led to a reliance on a number of techniques for isolating sensitive database information. These include physical protection, "system high" operation, and use of manual techniques for data sharing. These actions are very costly and detrimental to operational utility and flexibility.

Trusted Computing Bases (TCBs), such as Honeywell's LOCK[BOEB85b], have been designed to provide this type of control in terms of abstract entities and operations which reflect an operating system orientation. The LOCK security policy consists of a discretionary security policy and a mandatory security policy. The discretionary security policy enforces need to know structures, while the mandatory security policy provides a multilevel control policy. The multilevel control policy is a non-interference policy which addresses both access to data and the flow of information in the system.

A DBMS presents a more difficult security problem than that dealt with by current TCBs with their operating system orientation. This results from the ability of the DBMS to preserve or even enhance the information value of the data it contains. This is possible because it captures information in addition to the raw data values themselves through the incorporation of knowledge about the types of data and relationships among the data elements. A DBMS also allows for the creation of new data and relationships through the application of complex functions to the data. Because of these capabilities, one is forced to consider a number of factors beyond those normally addressed when dealing with operating system security. These include the impact of data context, aggregation, and inference potential.

1.2. Approach: Database Complementing OS

Honeywell's Lock Data Views (LDV), funded under contract F30602-86-C-0003 from the Distributed Systems Section/COTD of Rome Air Development Command, addresses the above problems by allowing individuals possessing a range of clearances to create, share, and manipulate databases containing information spanning multiple sensitivity levels. In LDV, the relational query language, Structured Query Language (SQL) [ASTR75], is enhanced with constructs for formulating security assertions. These security assertions serve to imply sensitivity labels for all atomic values, contexts, and aggregations in a database. The labelled data are partitioned across security levels, assigned to containers with dominating security markings or levels, and may only flow upward in level unless authorized otherwise. The ability of LDV to perform in this manner is a function of its design, and of the Operating System upon which it is hosted.

This paper describes one component of LDV, the update processor. Section 2 presents the security policy, section 3 presents an overview of the LDV system organization, and section 4 presents the design of the update processor.

2. Security Policy Overview

2.1. Security Policy Requirements

To meet the DoD security policy requirement, as stated in DoD Directives 5200.28 [DOD77], 5200.28-M [DOD79],a and 5200.1-R [DOD84], LDV must itself define a security policy that it enforces. The LDV security policy [HONE87] builds on the concepts of the LOCK security policy and extends them in a consistent and integrated fashion. The underlying LOCK security policy both constrains the actions of the DBMS and provides a foundation for the DBMS security policy. The latter provides extensions to the basic LOCK policy which respond to limitations in regards to database security concerns. The next subsection presents an overview of the basic LOCK security policy, followed by an overview of the DBMS policy requirements and extensions.

2.2. LOCK TCB Security Policy

The LOCK TCB satisfies the security requirements defined for the A1 level in the Trusted Computer Security Evaluation Criteria [DOD85]. These include requirements regarding mandatory and discretionary access control, object reuse, and maintenance, integrity, and export of sensitivity labels for objects, subjects, and devices. In addition, it supports the ancillary A1 requirements for accountability, audit, and assurance. The interested reader is referred to the Criteria [DOD85] for a discussion of these requirements.

The LOCK security policy at the highest level of abstraction states that:

"Data is labeled with a level and flows upward in level unless authorized to do otherwise."

This captures the DoD notion of security, which focuses on the confinement and protection of information (data in a context) from compromise.¹ The policy statement is interpreted in terms of a series of increasingly detailed specifications of the security relevant mechanisms for the system. This provides the basis for the enforcement of the security policy within the LOCK. Supporting mechanisms, such as user authentication and accountability, provide assurance that the security mechanisms act in a manner consistent with the security policy.

The type enforcement policy deals with aspects of security policy that are inherently non-hierarchical in nature. For example, payroll and medical records found in a database should probably not both be accessible by the same people. A full discussion of how and why the LDV design uses type enforcement to enforce protection within the database and between the database and other application domains is found in [HONE88], a short discussion of the concepts and mechanisms is found in [BOEB85a]. in terms of abstract entities and operations. There are three principal entities in the LOCK security policy: subjects, objects, and the Effective Access Matrix (EAM). Subjects are the active, process-like, entities in the system and objects are passive, file-like, entities. The EAM defines the permissible flows of information within the system. The EAM is computed based on the security relevant attributes associated with subjects and objects. The LOCK policy describes these attributes and the allowed accesses based on the notion of potential interferences between subjects.

In addition to the mandatory and discretionary security policies, LOCK provides labeling, integrity, authentication, and accountability mechanisms. These are described in [HONE87a].

2.3. DBMS Security Policy Requirements

The LOCK security policy is incomplete in dealing with DBMS security because of its operating system orientation. The most significant contributor to complexity within the DBMS environment is the information carrying potential of the database structure. The DBMS preserves or even enhances the information content of the database by incorporating knowledge of the types of data and relationships among the data. The data manipulation capabilities of the DBMS also allow the creation of new data and relationships through the application of complex functions to the stored data.

Our approach to providing a complete and tractable DBMS security policy extends the basic LOCK security policy through the incorporation of an explicit classification policy. The classification policy must address those factors which are crucial to a correct determination of the sensitivity level of data within the DBMS context. In particular, the policy considers:

Name-dependent classification: rules that refer to data items by name. This provides classification at the granularity of relations and attributes.

Content-dependent classification: rules that refer to the content of data item occurrences. This provides classification at the granularity of tuples and elements.

Context-dependent classification: rules that refer to combinations of data items. This can be used to reflect sensitivity of specific fields when accessed together.

Inference control: the determination of data sensitivity based on the potential inferences that can be made based on a sequence of access requests.

¹ Note that the simple-security and *- properties which form the basis of discussion of most security policies can be derived from the just-stated LOCK security policy.

2.4. DBMS Policy Extensions

The additional concern for a DBMS in a Multi-Level Secure environment beyond that of LOCK is the proper labeling of information. To provide for that concern, two extensions to the Policy of the TCB are required. One extension summarizes the actions that happen when a database is updated, and the other when a query is made to the database.

2.4.1. Update Classification Extension

For all security levels $L1$ and $L2$ ($L1 \leq L2$) and all base relations R in the database (where $L1$ is the basic level of R), a tuple T being stored securely in a partition P (at level $L2$) of R implies that the basic level of any of the data of T stored in P is $\leq L2$.

Definition: The $BASIC_LEVEL(T)$ of a tuple (or portion of a tuple) is the lowest level of the set of levels at which T can be securely stored. For a discussion, see [Honey87].

Informally, this means that we partition the data in the database by the base level security-level of the data. We use the enforcement of LOCK to provide most of the security, with the database extension mechanisms only handling special cases such as classification by context.

2.4.2. Response Classification Extension

For all responses R , and all objects O , a response R being written into object O implies that the security-level of the object O is in the group of levels defined by $Admissible_Derived_Level_Set(R)$.

Definition: The $Admissible_Derived_Level_Set(R)$ is the set of all levels for which releasing the information in the response R at that level will not enable any user to infer any further information whose sensitivity level exceeds that user's level. For discussion, see [Honey87].

Informally, this means that responses are written into ordinary objects (which afterwards can be shared in any arbitrary way, subject to normal system security constraints). The appropriate security level for the objects depends not only on the response, but upon what can be inferred by the response being released at that level.

3. Pipeline Organization

The way we enforce the two policies is by three assured pipelines. Assured pipelines originated in [Boeb85a], the pipeline integrity is itself enforced by the LOCK type enforcement mechanism. These pipelines pass through a number of subjects² in order to support encapsulation and the security and/or integrity policies. The three pipelines are:

- the query/response pipeline,
- the data input/update pipeline, and
- the database definition/metadata pipeline.

The first of these maps a query from the application domains to the DBMS, processes the query to produce a result relation, labels this result, and exports it to the user domain. This response pipeline runs untrusted in the early stages;³ the portion which determines the classification label of the data to be released is an example of a trusted portion.

The second pipeline allows subjects executing in a special data input domain to prepare records for input to the DBMS, identify records to delete, etc., and transforms them into a data type readable by the DBMS domain. This update pipeline also runs untrusted in the early stages; the portion which determines the data classification and where-to-write are trusted code.

² A detailed list of the subjects, together with the types of objects that they act upon, and the kind of accesses allowed by each kind of subject to those objects, is found in the complete report.

³ For example, the SQL parser can be untrusted design/code, since the worst it could do would be to create an internal form of a different SQL statement that the human user could enter externally anyhow.

The final pipeline provides the mechanism for defining a database structure specifying relations, views, attributes, classifications, etc., and would normally be restricted to access by the database administrator and by the database system security officer. As with the others, the metadata pipeline allows untrusted code in the early stages; an example of a trusted portion is that which actually stores the constraint metadata.

The remainder of this paper describes the update pipeline, the others are presented in [Hone88].

4. Update Pipeline

This section describes the design of the Update Pipeline of the LDV system. An introduction to the Update Pipeline, the major design issues considered in the design, and an overview of the design are presented. The detailed WELLMADE Design is presented in [HONE88].

4.1. Introduction

The Update Pipeline is the database updater for LDV. Update processing for LDV is complicated by the fact that the databases being managed are multilevel. In LDV, the relational query language, SQL, is enhanced with constructs for formulating security assertions. These security assertions serve to imply sensitivity labels for all atomic values, contexts, and aggregations in a database. The data are partitioned across security levels, assigned to containers with dominating security markings or levels, and may only flow upward in level unless authorized otherwise. The Assured Update Pipeline is a set of processes which execute multi-user update requests, distribute the data, and store the data at an appropriate level.

4.2. Design Issues

The major design issues are addressed in the Update Pipeline design: data distribution across files, and assuring the correctness of updates in spite of polyinstantiation (defined in Section 4.2.4). The issues are discussed in the following subsections.

4.2.1. Data Distribution

The first design issue is how to distribute multilevel data so that a large number of authorized users may obtain needed information from the data. This issue arises because it is not sufficient to simply store data into the right containers. It is also important that the method of storage of the data does not allow even authorized users at a given level to infer information at a higher level.

The LOCK TCB, whose security mechanisms are available to LDV, already enforces a security policy which stipulates that "Data may only flow upward in level unless authorized otherwise" [HONE87, pg. 62]. This policy is extended to LDV as follows: "Information may only flow upward in level unless authorized otherwise". In order to reduce the amount of trusted code in the DBMS, we have chosen to use the LOCK security mechanisms as much as possible and only augment those aspects that do not meet the requirements for information security. Since LOCK enforces its security policy to data stored in operating system files, LDV security assertions on data items must be transformed into LOCK security assertions on files. This transformation is carried out by the Update Pipeline, with assistance from the Metadata Pipeline.

As an example of the distribution problem for multilevel data, consider the following EMPLOYEE relation and associated assertions on its data values:

EMPLOYEE			
SSN	Name	Address	Salary
1	PD	St.Paul	100K
2	BT	Mpls	110K
3	EO	Mpls	110K
4	ON	Wash. DC	200K

Security levels are UNCLASSIFIED (U),
SECRET (S) and TOP SECRET (TS).

Constraints on EMPLOYEE are:

SSN is the key.

Default level for Name is (U).

Name is (TS) where Name = ON.

Default level for Address is (S).

Address is (TS) where Name = ON.

Default level for Salary is (S).

Salary is (TS) where Name = ON.

(Name, Salary) is (TS) when taken together.

One way to distribute this relation across LOCK data files is to use the method of [HINK75] and assign one file per attribute at the default level of the attribute, and then create an additional file for each additional level incurred by content-based security constraints. For the example under consideration, the following files are created:

Name-U	
1	PD
2	BT
3	EO

Name-TS	
4	ON

Address-S	
1	St. Paul
2	Mpls
3	Mpls

Address-TS	
4	Wash. DC

Salary-S	
1	100K
2	110K
3	110K

Salary-TS	
4	200K

The performance implication of the distribution scheme just described in which there is one file per attribute is unacceptable for a database of any size. The number of files that must be opened and joined within the response pipeline in order to reconstruct a view is too large - up to mn files per relation with m attributes and n possible sensitivity levels per attribute.

4.2.2. LDV Data Distribution Scheme

The basic scheme for data distribution across LOCK files is to assign a set of files per security level. There is no replication of data across levels. The Update Pipeline determines the appropriate assignment of data to files by examining the name-dependent, content-dependent, and context-dependent classification constraints. The view at any particular level is reconstructed by the *MERGE* operation of the response pipeline. Since partial relations that are stored at each level may have numerous null values, these nulls can be squeezed out by padding each partial tuple with a tuple descriptor. A tuple descriptor is a bitstring whose length is the order of the relation. A '1' in a position indicates that a value exists for that attribute, and a '0' indicates that the field is null. A 'D' in the first position indicates that the tuple has been logically deleted. In addition to the tuple descriptor, a timestamp and the level of the tuple are stored. The level of the tuple is the level at which the tuple was inserted. These three fields are not displayed to the user by default; they are manipulated internally by LDV.⁴ However, the user may request the retrieval of the timestamp and level fields. The tuple descriptor always precedes the tuple, followed by the timestamp, level, and values for the attributes that have '1's in their corresponding positions in the tuple descriptor. This scheme in conjunction with the File Manager that controls the opening of files eliminates the need to upgrade files that are involved in context-based constraints and consequently reduces the frequency of downgrades as well.

One scheme for data distribution is to upgrade the files involved in the context-based constraints, and, therefore, store the data involved at a higher level. This method is being used by SRI in the design of SeaView [SEAV88, pg. 6]. In the example being discussed, the files Name-U and Salary-S would be upgraded during update to TS. This alternative would tend to push all information up toward the maximal elements in the POSET of levels, and render the database virtually inaccessible to a majority of users. One way to avoid the inaccessibility problem is to design a Downgrader inside the query processor to downgrade information that can be accessed alone (approach used in Sea View). In the example, if a S-user or UNCLASSIFIED user (U-user) requested information concerning Name but not Salary, then information from the Name-U file would have to be downgraded. Unfortunately, this approach has a potential for accidentally downgrading different (say Rating-S) data if the downgrader process malfunctions.

4.2.3. Data Partitioning of Distributed Data

A closely related design issue to Data Distribution is data partitioning. Two kinds of data partitions that determine the type of reconstruction required are:

- (i) Replicate Lower Level Data in Higher Level Files.

This approach is advantageous in a retrieval-system with infrequent updates. A sequential scan property is achieved with minimal overhead since the synchronization of updates to the replicated data is not frequent. As an example, consider the files created for the EMPLOYEE relation in the previous example. The data in the Name-U file would have to be replicated for the S-users and again for the TOP SECRET users (TS-users); the data in the Address-S and Salary-S files would each have to be replicated for the TS-users.

The disadvantage of this approach is the complexity of the updates. In those few cases when data have to be updated, the security risks are non trivial. If there is a separation of physical media by level, then multilevel processes must be spawned and synchronized in order to update the replicated copies. In addition, the

⁴ To save space, the level is not actually stored in the tuple, but is derived from the level of the file when it is retrieved. It appears to LDV that the level was stored.

commit protocol requires each of these processes to signal success or failure back to some coordinating process resulting in a possible covert channel.

(ii) **Strict Partitions.**

Using this approach, lower level data are not replicated in the higher level files. A recovery algorithm is needed to reconstruct the partial relation representing a view at a given level. The advantage of this approach is the simplicity of updates. The disadvantage of this approach is the performance penalty for retrieval requests for the recovery algorithm.

In LDV, the data partitions are strict, with no replication; the query processor must first reconstruct the response at a level from fragments stored in various files. This reconstruction must be efficient. The LDV reconstruction algorithm is an efficient way to accomplish the LDV equivalent of a sequential scan of a relation. The reconstruction scheme does not preclude the use of access path selection strategies such as those developed in [SELI79] and which have been adapted for use in LDV (see WELLMAD design in [HONEY88]). This allows a more simple, more efficient update processor.

4.2.4. Assuring the Correctness of Responses In Spite of Polyinstantiation

Updates to multilevel data in the face of non-interference and non-disclosure policies may lead to "polyinstantiation" and inconsistencies. Polyinstantiation is an update anomaly which violates such basic integrity constraints as primary key constraints or, more generally, functional dependencies [DENN87]. For example, a U-user could inadvertently duplicate a primary key value that had been entered earlier by a S-user. Subjects must be shielded from such a phenomenon, by enforcing the basic integrity constraints on the result of a query. For flexibility, the user should be allowed to specify which tuples are to be filtered away from the response using time-oriented constructs and level-oriented constructs, as well as data definition constructs that allow a user to derive the values of attributes of one tuple from those of another. For example, the S-user may choose to see only those tuples that were entered after a certain time or to derive his/her own tuples from those entered by a CONFIDENTIAL user (C-user) rather than a U-user. The SQL language has been extended [HONEY88] in such a way. The preferred LDV approach to polyinstantiation is to allow for such flexibility as well as to enforce basic integrity constraints such as primary key constraints at each level⁵, functional dependencies, and multivalued dependencies. Initially, we require that primary keys and functional dependencies be enforced, meaning that the relations must be in Third Normal Form (3NF). Later, in order to support consistent Fourth Normal Form (4NF) relations, primary key constraints, functional dependencies, and multivalued dependencies may be stipulated and enforced.

4.3. Design Overview

This section presents an overview of the Update Pipeline design. Processing of an insert request, a delete request, a modify request, an overview of the major modules, and an overview of the security critical modules are presented.

4.3.1. Insert Request

An insert request must be processed so that data is inserted into the correct file at the correct level based upon the classification constraints and the inserting subject's level. Upgrades are determined by the values of the elements of the tuple to be inserted.⁶

The insert request will first be imported into the DBMS domain. The imported request will be sent to an upgrader which will compute the level of the insert operation as follows:

1. The level of each attribute specified in the insert request is set to the corresponding default level.

⁵ Primary keys are enforced per level and not across levels thus allowing for polyinstantiation.

⁶ The alternative is to have the Response Pipeline do the upgrade when data are retrieved. We believe that the retrieval alternative requires more trusted code.

2. The relevant constraints visible at the processing subject's level are retrieved. Each relevant constraint satisfies the following conditions:

it classifies an attribute which is specified in the insert request at a level which dominates the processing subject's level, and it has not been examined during a previous iteration of this algorithm.

3. For each relevant constraint a new level is computed for each attribute that is classified by the constraint as follows:

new level = least upper bound(old level, level specified in the constraint).⁷

4. Compute the least upper bound of the levels of the processing subject and all the attributes specified in the insert request.

5. If the new level dominates the level of the processing subject, then create a new subject at this new level and pass the parameters associated with the current processing subject to the new subject. Delete the current processing subject. The new subject becomes the current processing subject. Go back to step 2.

6. Otherwise, if the new level is equal to the current processing subject's level, then continue with the remaining processing of the insert operation, i.e., this is the level of the insert operation.

We will illustrate this algorithm with a simple example. Let $R(A1,A2,A3)$ be a relation with the following constraints:

C1: If $A2 = 5$ then $A1$ is TS.

C2: If $A3 = ttt$ then $A2$ is S.

C3: C1 is S.

C4: C2 is U.

C5: Default level of $A1, A2, A3$ is U.

A U subject requests to insert (alpha, 5, ttt) into R. Initially the processing subject's level is U and the default levels of all three attributes are U. During the first pass of the upgrader, the relevant constraint is C2. The level of the attribute A2 is computed to be S. Then the new level will be set to be the least upper bound of the levels of A1, A2, A3 and the processing subject's level. This new level is S. A new processing subject will be created at the secret level. During the second pass the relevant constraint is C1. The level of A1 will be computed to be TS. The new level will be the least upper bound of the levels of A1, A2, A3, and the processing subject's level. This new level is TS. A new processing subject will be created at the TS level. During the third pass no relevant constraints will be retrieved. The levels of the attributes remain the same. The new level will be computed to be TS. This new level is the same as the processing subject's level. Therefore the insertion will be performed at the TS level.

After the level of the insertion is computed, a view of the relation specified in the insert request is built using the *MERGE* operation of the Response Pipeline. Once the view is built the request may be modified if necessary as follows:

If the primary key value specified in the request already exists and the tuple is only visible at the level of the insert operation and not below this level, then the request is rejected as it is a duplicate tuple with the same primary key. If it is not a duplicate tuple at the level of the insert operation, then the tuple is inserted with a new timestamp and the level of insertion into a file at the level of the insert operation.

The modified request is optimized and subsequently an execution strategy is generated. That is, the requests on relations are translated into requests on files. In the previous example the request will be translated into operations on a TS file, say F1, as follows:

OPEN F1

⁷ We assume that the set of levels forms a lattice.

INSERT (alpha, 5, ttt) INTO F1
 CLOSE F1

Information about the file F1 will be retrieved from the Data Dictionary. This is because the data dictionary includes the association between the file F1 and the relation R.

We will illustrate the insert operation with some examples.

Consider a relation R(A1,A2,A3) with the following constraints:

A1 is primary key

A1 is TS if A2 = 5

A2 is TS if A2 = 5

A3 is TS if A3 = www or yyy

default level of A1, A2, A3 is S

level of the constraints is U

The relation R will be stored in a S file F-S and a TS file F-TS. Recall that the first field is the tuple descriptor, the second field is the timestamp, and the third field is the level.

F-S					
111	00	S	alpha	17	xxx
110	01	S	beta	34	
111	02	S	delta	20	uuu

F-TS					
101	01	TS	beta	www	
111	03	TS	gamma	5	yyy

The views at levels S and TS are V-S and V-TS respectively⁸. They are computed using the *MERGE* operation in the Response Pipeline. Note that the timestamp and level proceed each attribute.

V-S									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
110	01	S	beta	01	S	34			
111	02	S	delta	02	S	20	02	S	uuu

⁸ The tuple descriptors, timestamps, and levels are not displayed to the user, but are shown here for illustrative purposes.

V-TS									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	01	TS	beta	01	S	34	01	TS	www
111	02	S	delta	02	S	20	02	S	uuu
111	03	TS	gamma	03	TS	5	03	TS	yyy

In the examples below it is assumed that the level of the insert operation has already been computed.

Example 1:

Suppose that a S subject requests to insert (gamma, 22, zzz).

This example illustrates the case where a subject attempts to insert new data where data already exist with the same primary key at a higher level. The solution is to insert the tuple at level S with a timestamp and level. The primary key and level uniquely identify the tuple.

After the insertion, the file F-TS does not change. F-S, V-S, and V-TS are changed as follows:

F-S					
111	00	S	alpha	17	xxx
110	01	S	beta	34	
111	02	S	delta	20	uuu
111	04	S	gamma	22	zzz

V-S									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
110	01	S	beta	01	S	34			
111	02	S	delta	02	S	20	02	S	uuu
111	04	S	gamma	04	S	22	04	S	zzz

V-TS									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	01	TS	beta	01	S	34	01	TS	www
111	02	S	delta	02	S	20	02	S	uuu
111	03	TS	gamma	03	TS	5	03	TS	yyy
111	04	S	gamma	04	S	22	04	S	zzz

Example 2:

Let F-S, F-TS, V-S, V-TS contain the values at the end of Example 1.

Suppose that a TS subject requests to insert (alpha, 18, aaa).

This example illustrates the case where there is a tuple at the lower level with the same primary key. The solution is to insert the tuple at the higher level with a new timestamp and level.

After insertion only F-TS and V-TS change as follows:

F-TS					
101	01	TS	beta	www	
111	03	TS	gamma	5	yyy
111	05	TS	alpha	18	aaa

V-TS									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	01	TS	beta	01	S	34	01	TS	www
111	02	S	delta	02	S	20	02	S	uuu
111	03	TS	gamma	03	TS	5	03	TS	yyy
111	04	S	gamma	04	S	22	04	S	zzz
111	05	TS	alpha	05	TS	18	05	TS	aaa

Example 3:

Let F-S, F-TS, V-S, V-TS contain the values at the end of Example 2.

Suppose that a TS subject requests to insert (pi, 10, bbb).

This example illustrates the case where a subject attempts to insert a tuple and no tuple exists in the database with the same primary key. The solution is to insert the tuple into F-TS.

After insertion only F-TS and V-TS change as follows:

F-TS					
101	01	TS	beta	www	
111	03	TS	gamma	5	yyy
111	05	TS	alpha	18	aaa
111	06	TS	pi	10	bbb

V-TS									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	01	TS	beta	01	S	34	01	TS	www
111	02	S	delta	02	S	20	02	S	uuu
111	03	TS	gamma	03	TS	5	03	TS	yyy
111	04	S	gamma	04	S	22	04	S	zzz
111	05	TS	alpha	05	TS	18	05	TS	aaa
111	06	TS	pi	06	TS	10	06	TS	bbb

Example 4:

Let F-S, F-TS, V-S, V-TS contain the values at the end of Example 3.

Suppose that a TS subject requests to insert (pi, 10, kkk).

This example illustrates the case where a subject attempts to insert a tuple and there is already a tuple with the same primary key at the same level. The solution is to reject the insert request.

4.3.2. Delete Request

Processing of a delete request is less complex than that of an insert request. In this case it is not necessary to compute the level of the delete operation as it is assumed to be that of the processing subject. This is because the *-property enforced by LOCK prevents higher level subjects from deleting information from lower level files. Therefore, upgrading the level of the delete operation does not make sense. A delete request is first imported. Then a request is made to the Response Pipeline to build a view of the relation specified in the delete request at the level of the processing subject's level. The delete request is modified according to the view just built as follows:

1. For each tuple being deleted, if any part of the tuple is visible at a lower level, then the delete request is rejected. This is because a higher level subject cannot write into a lower level file.
2. If the subject wants to delete the portion of the tuple visible at its level, then the values corresponding to this portion are changed to NULL.
3. If no part of the tuple to be deleted is visible at a lower level, then the tuple is marked as deleted in the file at the level of the delete operation. The tuple is not removed from the file immediately because it may be required by a higher level subject in reconstructing the higher level view using MERGE. An expunge daemon will periodically review the files and remove the tuples that are marked as deleted. Before removing the tuples, the daemon will insert them into the appropriate higher level files. We expect that the expunge daemon will actually be a set of subjects running at various levels under the control of the DBSSO. They would look at tuples logically deleted over some time period (for example, 1 month) and do the physical deletion.

Example 5:

Let F-S, F-TS, V-S, V-TS contain the values at the end of Example 4.

Suppose that a S subject requests to delete the tuple where A1 = 'beta'.

This example illustrates the case where a lower level subject tries to delete a tuple that is used in building a view at a higher level. The solution is not to remove the tuple but to mark it as deleted⁹. After the delete operation, no changes will be made to F-TS and V-TS. F-S and V-S will be changed as follows:

F-S					
111	00	S	alpha	17	xxx
D110	01	S	beta	34	
111	02	S	delta	20	uuu
111	04	S	gamma	22	zzz

V-S									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	02	S	delta	02	S	20	02	S	uuu
111	04	S	gamma	04	S	22	04	S	zzz

4.3.3. Modify Request

The modify request is treated as a delete request followed by an insert request. Therefore the details will not be described. We will illustrate the modify request with an example.

⁹ The D in the first column is a flag that marks a logically deleted tuple. Where no D is shown the delete-flag is present, but is not enabled and will not be shown.

Example 6:

Let F-S, F-TS, V-S, and V-TS contain the values at the end of Example 5.

Suppose that a TS subject requests to modify A2 = 81 where A1 = delta.

This example illustrates the case where a subject attempts to modify an element with a lower access class. The solution is to insert the tuple with a different timestamp and level at the higher level. (Note that the lower level information cannot be deleted due to the *-property). The file F-S will not change as a result of the modify operation. F-TS and V-TS will be changed as follows:

F-TS					
101	01	TS	beta	www	
111	03	TS	gamma	5	yyy
111	05	TS	alpha	18	aaa
111	06	TS	pi	10	bbb
111	07	TS	delta	81	uuu

V-TS									
TD	T	L	A1	T	L	A2	T	L	A3
111	00	S	alpha	00	S	17	00	S	xxx
111	01	TS	beta	01	S	34	01	TS	www
111	02	S	delta	02	S	20	02	S	uuu
111	03	TS	gamma	03	TS	5	03	TS	yyy
111	04	S	gamma	04	S	22	04	S	zzz
111	05	TS	alpha	05	TS	18	05	TS	aaa
111	06	TS	pi	06	TS	10	06	TS	bbb
111	07	TS	delta	07	TS	81	07	TS	uuu

4.3.4. Overview of the Major Modules

The major modules in the Update Pipeline are the User Request Manager (URM), the Relational Access Manager (RAM), and the Execution Manager (EM). The relationship between these major modules is shown in Figure 1. Each major module will be discussed below.

4.3.4.1. User Request Manager

The URM provides an SQL interface to LDV for updates that is consistent with the ANSI SQL standard. It initially performs discretionary access control on views as defined by the ANSI standard (i.e., not LOCK's discretionary access control policy), update modification of updates on views to form updates on base relations, integrity checking, and classification constraint enforcement. All information needed for translation is provided by the Data Dictionary Manager (DDM). Of particular interest is the update security modification and insert-level calculation.

The Update Security Modification modifies the update request using the classification constraints. For an insert request it computes the level of the insert using the upgrader, builds the view of the relation being updated using *MERGE*, and checks for a tuple with the same primary key visible at the level of the insert and not below (it rejects the insert if it finds one). The Upgrader determines the level of an insert using classification constraints. If the a predicate of a constraint evaluates to TRUE, it is used to assign a new level, otherwise, the constraint is ignored. The output is the modified insert request and its level.

For a delete request, it builds the view of the relation being updated using *MERGE*, builds a list of tuple identifiers (timestamp, level, primary key) being deleted by eliminating those that are visible at a lower level, and

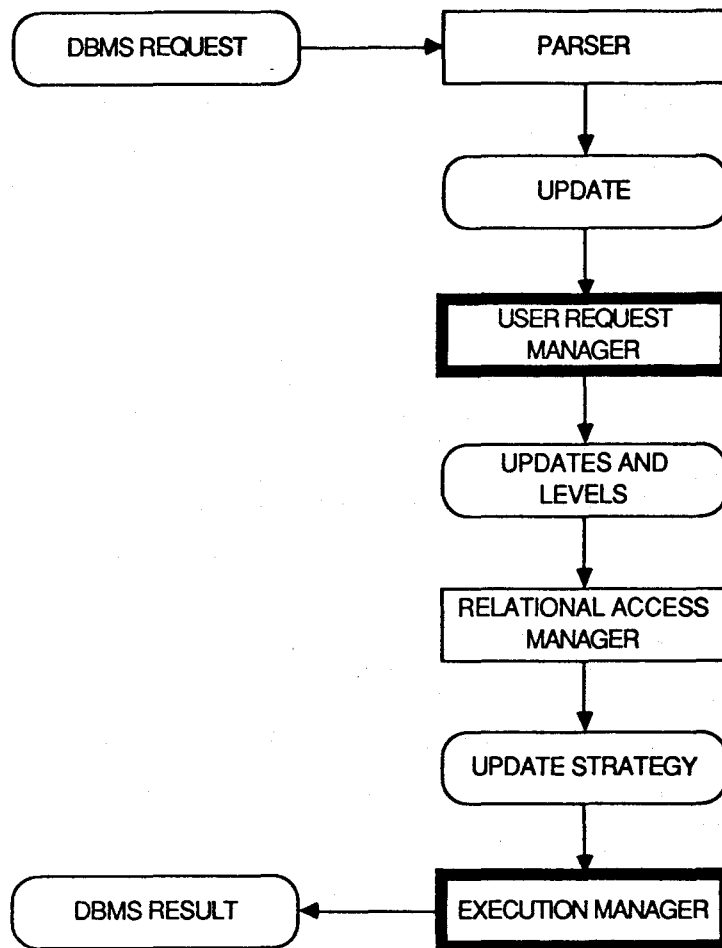


Figure 1: Update Pipeline

builds a delete request that deletes those tuples. The output is the delete request and its level.

For a modify request, it does the delete processing followed by the insert processing described above. The output is a delete request and its level, and an insert request and its level.

4.3.4.2. Relational Access Manager

The RAM takes the internal representation of the update built by the URM, and builds an optimal execution strategy. The information on access paths required by the optimization process are obtained from the DDM.

4.3.4.3. Execution Manager

The EM takes the execution strategy produced by the RAM, and executes each operation in the strategy using the services of the Relational File Manager (RFM). The RFM is broken up into the following managers:

- relation manager;
- record manager;
- index manager; and
- file manager.

The relation manager manages relations by using the services of the other data storage and retrieval managers. The record manager manages collections of records stored as tuples by the relation manager. The record manager uses the services of the file and index managers to store and retrieve records. The index manager is used by the relation manager to store keys in an index and manage them in sorted order. The index manager uses the services of the file manager to store and retrieve keys. The file manager manipulates the data management files.

4.3.5. Security Critical Modules

In LDV, we restrict security-critical code to a subset of the modules. Because of the organization of LDV as subjects performing designated controlled roles, and objects touchable only by certain role-players, only those modules executed by subjects that compute security-critical information (such as the level of an insert) or that touch security-critical data (such as a file containing the database data) are security-critical. Details of how these modules are assigned to subjects and how LOCK functions are used by these modules are given in [HONE88]. The LDV FTLS will model the security critical modules and show how these concerns are addressed.

5. Conclusion

Given the additional problems introduced by increased granularity of items in a database over files, the possibilities of inference and aggregation, and the need to manage metadata as well as data in a secure way, the way in which to design and organize a secure database is not obvious.

In this paper we have shown the need for a security policy for a database system that builds upon the classical security policies for operating systems. We have described our policy for LDV and shown how it builds on the policy for LOCK. We have described some of the problems associated with multi-level databases and our approach to solving them. We have described how our pipeline organization helps to minimize the amount of design and code that must be trusted and/or verified.

Our complete report[Hone88] describes additional challenges we faced, and the approach we are taking towards solving them.

We look forward to describing our implementation at a future date.

6. References

- [AFSB83] Committee on Multilevel Data Management Security. "Multilevel Data Management Security", Air Force Studies Board, National Research Council, National Academy Press, 1983. For Official Use Only.
- [ANSI86] American National Standards Institute, ANSI Standard X3H2-86-2, ANSI SQL, January 1986.
- [BELL25] Bell, D.E., and L.J. LaPadula, "Secure Computer System: Unified Exposition and Multics Interpretation", Tech. report MTR-2997, MITRE Corp., July 1975.
- [BIBA77] Biba, K.J., "Integrity Considerations for Secure Computer Systems", ESD-TR-76-372, USAF Electronic Systems Division, Bedford, Mass., April, 1977.
- [BOEB85a] Boebert, W. E., Kain, R. Y., "A Practical Alternative to Hierarchical Integrity Policies." Proceedings of the 8th Annual National Computer Security Conference, NBS, 1985, pp. 18-27
- [BOEB85b] Boebert, W. E., Young, W. D., Kain, R. Y., Hansohn, S. A., "Secure Ada Target: Issues, System Design and Verification", 1985 IEEE Symposium on Security and Privacy, Oakland, California, April, 1985.

- [BOEB86] Boebert, W.E., Dillaway, B.B. and Haigh, J.T., "Mandatory Security and Database Management Systems", Proceedings of the National Computer Security Center Invitational Workshop on Database Security, June 1986.
- [CLAY82] Claybrook, Billy G., and Epstein, Harvy I., "A Security Model for a Multilevel Secure Database Management System", The MITRE Corporation, Bedford, Massachusetts, M82-12, February 1982.
- [CODD70] Codd E.F., "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, Vol. 13, No. 6. June 1970, pp. 377-387.
- [DATE83] Date, C. J., "An Introduction to Database Systems", Second Edition, Addison-Wesley, London, England, 1983.
- [DENN87] Denning, Dorothy et al, "A Multilevel Relational Data Model", 1987 IEEE Symposium on Security and Privacy, Oakland, California, 1987.
- [DENN87b] Denning, Dorothy et al, "On the Completeness and Consistency of Security Constraints", 1987 IEEE Symposium on Security and Privacy, Oakland, California, 1987.
- [DILL86] Dillaway, B.B. and J.T. Haigh, "A Practical Design for a Multi-Level Secure Database Management System," Proceedings of the Second Aerospace Computer Security Center Invitational Workshop on Database Security, June 1986.
- [DOD77] "Security Requirements for Automatic Data Processing (ADP) Systems", Department of Defense Director Number 5200.28, May 6, 1977.
- [DOD79] "ADP Security Manual", Department of Defense Manual Number 5200.28M, June 25, 1979.
- [DOD84] "Information Security Program Regulations", Department of Defense Number 5200.1R, October 2, 1984
- [DOD85] "Trusted Computer System evaluation Criteria", Department of Defense Standard 5200.28-STD, December 26, 1985.
- [DOWNS86] Downs, D.D., "Discretionary Security in Database Management Systems," Proceedings of the National Computer Security Center Invitational Workshop on Database Security, June, 1986.
- [DWYE87] Dwyer, P.A., Jelatis, G.D., Thuraisingham, B. M., "Multi-level Security in Database Management Systems", Computers and Security, North-Holland, 1987.
- [GALL78] Gallaire, H., and Minker, J., "Logic and Databases", Plenum Press, 1978.
- [HENN87] Henning, R.R. et al., "Computer Architectures, Database Security, and an Evaluation Metric," Proceedings of the Third International Conference on Data Engineering, Feb., 1987, pp. 518-527.
- [HINK75] Hinke, T.H., and Schaefer, M., "Secure Data Management System", Technical Report RADC-75-266, Systems Development Corporation, November, 1975.
- [HONE79] Kelly, B.K., Joy, R., Devor, C, and Stachour, P., "the WELLMAD system design methodology", Honeywell Corporate Computer Sciences Center, Bloomington, Minnesota, 1979.
- [HONE87] "Secure Distributed Data Views - Security Policy Extensions", Interim Report A002 prepared for the Rome Air Development Center, Contract F30602-86-C-0003, Honeywell Systems Research Center and Corporate Systems Development Division, April, 1987.
- [HONE87a] "B-Level Design Specification for the LOCK Operating System", CDRL A009, Contract MDA 904-87-C-6011, Honeywell Secure Computing Technology Center, St. Anthony, Minnesota, June 1987.

[HONE88] "Secure Distributed Data Views - Implementation Specifications", Interim Report A005 prepared for the Rome Air Development Center, Contract F30602-86-C-0003, Honeywell Systems Research Center and Corporate Systems Development Division, May, 1988.

[MILL84] Millen, J.K., Cerniglia, C.M., "Computer Security Models:", Mitre Technical Report MTR9531, September 1984.

[NCSC84] Proceedings of the National Computer Security Center Invitational Workshop on Database Security, Baltimore, Maryland, June 17-20, 1986.

[SEAV88] Seaview Project, Lunt, Teresa F., "The SEAVIEW Formal Top Level Specifications", Report A004 on Contract F30602-85-C-0243, SRI International, Menlo Park, California, 1988.

[SELI79] Pat Griffiths Selinger et al, "Access Path Selection In a Relational Database Management System", Proceedings, ACM SIGMOD, Boston, Massachusetts, May 30 - June 1, 1979.

[ULLM82] Ullman, J. D., "Principles of Database Systems", Computer Science Press, Rockville, Maryland, 1982.

[YOUN86] Young, W. D., Telega, P. A., Boebert, W. E., and Kain, R. Y., "A Verified Labeller for the Secure Ada Target", Proceedings of the 10th Annual National Computer Security Conference, NBS, 1986.



**ORDERING INFORMATION FOR
IBM SECURITY FILMS**

IBM Corporation offers two security awareness video tapes, "An Ounce of Prevention" and "Information - Handle with Care." "An Ounce of Prevention" is 11.5 minutes long and discusses office systems security. "Information - Handle with Care" runs 10 minutes and addresses the protection of information and information systems. Both films are available in 3/4" videotape, 16mm film, and 1/2" VHS and Beta formats.

Copies may be obtained via loan from:

Modern Talking Picture Service
5000 Park Street North
St. Petersburg, Florida 33709
(813) 541-7571

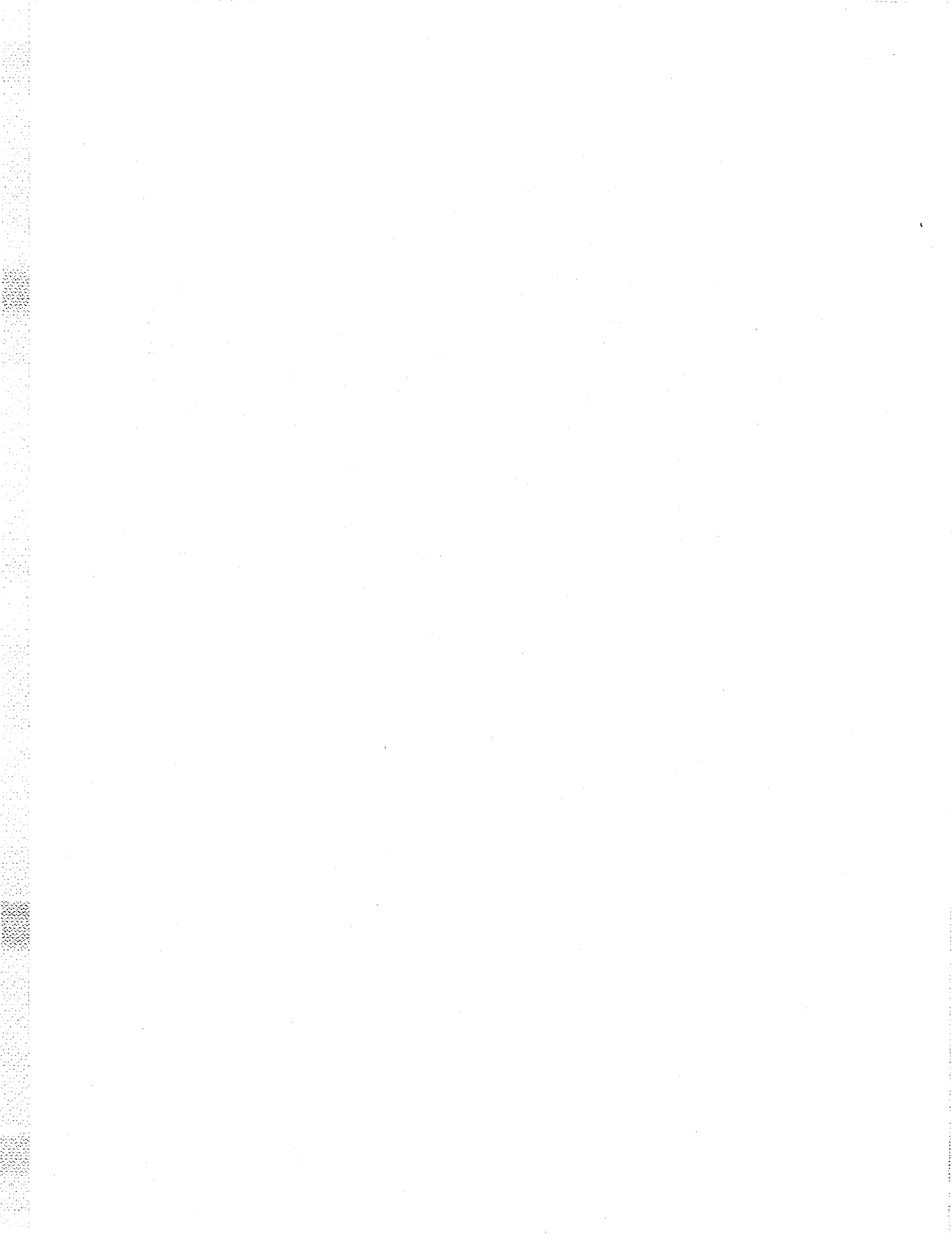
They may be purchased from:

Glyn/Net
356 West 58th Street
New York, New York 10019
(212) 560-6980

Current pricing information may be obtained from the vendor.

If you have questions about these films, please feel free to contact:

William L. Davis, Jr.
Program Director, Data Security Programs
IBM Corporation
2000 Purchase Street
Purchase, New York 10577



NATIONAL COMPUTER SECURITY CONFERENCE

PLENARY SESSION

1030-1200

TUESDAY, OCTOBER 18, 1988

THE COMPUTER SECURITY ACT OF 1987

A FOCUS ON HOW NIST/NSA WILL INTERACT
ON POLICY, IMPLEMENTATION, AND TECHNOLOGY

MODERATOR: MR. STEPHEN WALKER, TRUSTED INFORMATION SYSTEMS

PARTICIPANTS:

MR. PATRICK R. GALLAGHER, JR., DIRECTOR, NCSC
MR. JAMES BURROWS, DIRECTOR, NIST-NCSS
MR. ELIOT SOHMER, NCSC
DR. DENNIS BRANSTAD, NIST
MR. JERRY RAINVILLE, NSA
DR. STUART KATZKE, NIST

PUBLIC LAW 100-235
IMPACT ON AND IMPLICATIONS FOR NIST and NSA

A plenary session of the 11th Annual National Computer Security Conference was held on Tuesday, 18 October 1988, to discuss policy, implementation, and technology issues stemming from the enactment of the Computer Security Act of 1987. Participants in the session were: Moderator, Mr. Stephen Walker, Trusted Information Systems; Mr. Patrick Gallagher, Jr., Director of the National Computer Security Center at the National Security Agency; Mr. James Burrows, Director of the National Computer Systems Laboratory (NCSL) at the National Institute of Standards and Technology; Mr. Eliot Sohmer, Chief, Office of Computer Security Evaluations, Publications, and Support at the National Computer Security Center; Dr. Dennis Branstad, Senior Fellow, NCSL; Mr. Jerry Rainville, Chief of Domestic Affairs for the National Security Agency; and Dr. Stuart Katzke, Chief, Computer Security Division at the NCSL.

The moderator prepared background information on the Computer Security Act, and asked questions of the participants concerning their respective and cooperative roles in implementing the legislation. Those questions and answers are included in this publication.

In addition, the session attendees were invited to submit written questions for the panel. Because there was not sufficient time for the panel members to answer all of the submitted questions, answers to those questions are included in Section II of this publication. The responses represent the coordinated thinking of both NIST and NSA at the time. Other supplemental information referred to in the answers to the questions is included in Appendices A and B.

TABLE OF CONTENTS

Section 1. Prepared Questions and Responses

I. Background	122
II. Public Law 100-235	124
III. NSDD-145 Impact	128
IV. Trusted System Technology	131
V. Cryptography	137
VI. INFOSEC Products	142
VII. Trusted Network Interpretation/Trusted Database Interpretation	144
VIII. Trusted UNIX Activities	146
IX. International Security Standards, Export Control	148

Section 2. Questions from the Floor and Combined NIST/NSA Responses	150
---	-----

Section 3. APPENDICES

A. PL 100-235, Computer Security Act of 1987	166
B. Warner Amendment	174

SECTION 1. PREPARED QUESTIONS AND RESPONSES

I. BACKGROUND

I.1. Mr. Walker: The purpose of the Public Law is to assign to the National Bureau of Standards [now the National Institute for Standards and Technology (NIST)] responsibility for developing standards and guidelines for Federal computer systems, including standards and guidelines needed to assure the cost-effective security and privacy of sensitive information in Federal computer systems, drawing on the technical advice and assistance (including work products) of the National Security Agency, where appropriate.

It further defines "sensitive information" to mean "any information, the loss, misuse or unauthorized access to or modification of which could adversely affect the national interest or the conduct of Federal programs, or the privacy to which individuals are entitled under (the Privacy Act), but which has not been specifically authorized under criteria established by an Executive Order or an Act of Congress to be kept secret in the interest of national defense or foreign policy."

I believe all of that to mean that NIST is responsible for developing security and privacy standards for Federal computer systems handling unclassified but sensitive information, with a few approved exceptions. Do you agree with this statement?

Mr. Burrows - Yes
Mr. Gallagher - Yes

I.2. Mr. Walker: I further believe that by implication the Public Law leaves to existing authorities the responsibility for handling sensitive information which has been authorized to be kept secret. Generally with respect to classified information and information that falls under the provisions of the Warner Amendment exemption, the provisions of NSDD-145 and any other Executive Orders or Presidential directives apply; and thus under NSDD-145, NSA retains its responsibilities with respect to classified information. Do you agree with this statement?

Mr. Burrows - Yes
Mr. Gallagher - Yes

BACKGROUND

I.3. Mr. Walker: The provision of the Public Law regarding NIST "drawing on the technical advice and assistance (including work products) of the National Security Agency, where appropriate" to "assure the cost effective security and privacy of sensitive information on Federal computer systems" seems to recognize that both NSA and NIST are intended to have significant efforts in this area and are to work together for the common good. Do you agree with this statement?

Mr. Burrows - Yes

Mr. Gallagher - Yes

II. PUBLIC LAW 100-235

II. 1. Mr. Walker: Public Law 100-235 has been in effect since January 8, 1988. What activities have occurred to date at NIST towards implementing this law?

Mr. Burrows: Since the law passed in January of this year, we have made several changes to our program. The activities required by the law fall into five categories: (1) computer security training; (2) computer security planning; (3) establishment of the Computer Security and Privacy Board; (4) cooperative activities; and (5) technical activities. I will outline the progress and status in each area.

We have worked closely with the Office of Personnel Management to develop a training plan for the U.S. Government. The plan was issued as a Federal regulation on July 13, 1988. We have developed several security awareness training guides to be issued in 1988 for senior executives, managers, and users of computers.

We have collaborated with OMB and NSA in the development of Bulletin No. 88-16, which provides guidance for the preparation and submission of security plans. We co-sponsored two workshops on training and planning that were held at NIST to help agencies in their training and planning activities required by the law.

The Department of Commerce has approved the charter for the Computer System Security and Privacy Advisory Board established under the law. Selection letters for the twelve members have been prepared and we expect to hold the first meeting around the end of this year.

During this year, we have continued most of our computer security program based on the foundation of technical, management, physical and administrative security standards and guidelines that we have developed and issued since the mid-1970's. Because of the added responsibilities of the law, we have deferred or delayed our activities in risk management, audit trail collection/analysis, POSIX security, and data integrity. Because of contractual obligations to other agencies, we continued our work on OSI security, electronic certification and "Smart Card" technology.

In the area of cooperation, we have worked with OMB, GAO, OPM and several other agencies to carry out the requirements of the legislation. We have worked very closely with NSA. Since passage of PL 100-235, we have called upon NSA and the National Computer Security Center (NCSC) to: (1) help implement the

PUBLIC LAW

security plan evaluation process; (2) assist in developing criteria for system integrity and availability that would augment the present computer security criteria; (3) ensure the continued availability of "C2" systems for protecting the confidentiality of data in shared systems; (4) develop guidance on the application of Trusted Computer Systems in unclassified, but sensitive, information processing; (5) develop standards for security in OSI and ISDN; (6) develop a family of security algorithms and systems that are exportable, publicly available, and NIST-supported, and that provide commercial levels of security; and (7) develop a technology base for personal security devices that are portable, exportable, and publicly available.

II. 2. Mr. Walker: Please summarize the areas of cooperation between NIST and NSA before and since the passage of the Public Law.

Mr. Gallagher: There have been many cooperative activities between NIST and NSA, both before and after the passage of the Public Law. Jim Burrows has identified many of them. I'll highlight some and add a few of my own. We have jointly established a Risk Management Research Laboratory housed at the NIST facilities in Gaithersburg, MD, to provide user support in risk management and a joint agency R&D facility for investigating risk management techniques. We will use the lab to further advance the state of the art in risk management while the NCSC supports and augments that research with concentration on risk management aimed at the challenges of multilevel security systems and the aggregation of classified information. The two agencies have also jointly hosted a risk management model builders' workshop for the purpose of developing a framework under which risk management methodology would fit. A second invitational workshop is planned for June 1989. We continue to cooperate closely in assessing various candidate methodologies for conducting automated risk management. When further research on new and emerging methods of risk management are available, future joint workshops will be sponsored. We are also cooperating on the effort to develop definitions for integrity. Once there is general agreement in this area, the process of defining criteria for this aspect of the security problem will begin. The SDNS project is another important area in which we work closely. Also, in the R/D area, NIST is now a participant in our extended technical review group.

II. 3. Mr. Walker: What is the status of the Memorandum of Understanding between NSA and NIST regarding respective roles in computer security?

Mr. Burrows: NSA and NIST have each drafted several Memoranda of Understanding but agreement on an MOU has not yet been

PUBLIC LAW

reached. During the Congressional hearings on September 22 of this year, the MOU was discussed by NSA, NIST, and the Congressmen conducting the hearings. It was clear that everyone wanted an MOU but several difficult issues were still unresolved. In particular, the question of the NIST role in cryptography as it is used in protecting information in computer networks was raised. A second issue raised at the hearing concerns the process to be followed if the parties involved do not agree that a security standard, especially one that involves cryptography, should be issued by the Secretary of Commerce. The Director of NSA stated that an MOU would be reached in the near future, that is, within the next quarter (three months). The Deputy Director of NIST stated that the MOU would have to include agreements on these sensitive areas in order to be useful.

Mr. Gallagher: I believe NIST has expressed the status of this activity accurately.

II. 4. Mr. Walker: What resources (people and money) has NIST been given or anticipate receiving to meet the expanded responsibilities under PL 100-235?

Mr. Burrows: NIST had a Congressional budget of \$1M for computer security in FY88 but was given no additional resources to implement PL 100-235 after the bill was passed. The activities required by the new law were performed by deferring or delaying several standards development activities that were under way. Of the \$5 million increase that was estimated as being required by NIST to perform the activities required by the law, the Executive Branch requested a \$3 million increase to NIST's base program from Congress in FY89. The budget that was recently passed cut this to a \$1.5 million increase for the NIST security program.

The additional money for NIST will support an additional ten technical staff plus support people and equipment. This will bring our total staff to 25. The Congressional budget request includes a \$3M increase for FY90 and a \$5M for FY91. These increases would support needed research and development activities as well as needed standards activities.

II. 5. Mr. Walker: What activities do you plan to perform in FY89 with these resources?

Dr. Katzke: The \$1.5 million will be used to support the Computer Systems Security and Privacy Advisory Board, the system security planning and review process, and the training and awareness activities that are required under the Act. We will increase our efforts in areas that were either slowed down or deferred in 1988 such as system and data integrity, POSIX

PUBLIC LAW

security, risk management, and audit trail collection and analysis. Depending upon available resources, we may start new activities in the areas of digital signatures, guidance on the use of trusted systems for protecting unclassified data, augmentation of the Orange Book criteria with integrity and availability requirements, security controls during the system development life cycle, and revision of FIPS 140 (i.e., FS 1027).

II. 6. Mr. Walker: What are your plans to review and comment on all of the security plans that agencies must submit by January 8, 1989?

Dr. Katzke: Based on a GAO report on the number of systems in the Federal government that are processing sensitive but unclassified information (60,000), reviewing and commenting on all the plans expected to be submitted by January 8, 1989, will be a formidable task. Because this is a requirement of the Law, we are expending a great deal of effort in planning for the review process. NIST will allocate several people to coordinate and manage the review with the NCSC. We are working with agencies to prepare plans that are complete and adequate, which will then reduce the number of comments that will be made.

We have already spent a great deal of effort in aiding agencies in developing the plans. We have held two workshops for the agencies on how to develop security plans and security training programs. Nearly 700 attended the first and over 500 attended the second, held at NIST on October 13 and 14. In the future, we plan to automate the planning process, the plan submission process and perhaps even the plan evaluation process. We hope that from this evaluation process, the need for common solutions, including standards, to computer security problems should become clear to all concerned.

Mr. Sohmer: We expect the plans submitted in the first round to be diverse in content and style, and analysis of them will be time-consuming, but we hope they reveal a great deal about the Government's security posture. We are committed to working together on this initiative. We will learn from it and go to "second round" planning from there.

III. NSDD-145

III. 1. Mr. Walker: What impact does passage of Public Law 100-235 have on NSDD-145?

Mr. Gallagher: The Law made explicit where NIST and NSA authorities and responsibilities lie. Public Law 100-235 makes clear that NIST is to lead in the protection of unclassified sensitive information. NSA continues to lead in the protection of classified information and information pertaining to systems covered by the Warner Amendment. NSDD-145 is still officially in effect and many of its mechanisms are very useful. Much good has come out of the NTISS structure and its two subcommittees. We believe that some structure that is broadly based across the Government should continue. Examples of valuable NTISS mechanisms include: (1) having a forum in which all of Government can assemble to discuss approaches to solving their INFOSEC problems; and (2) the NTISSC publication system--specifically, NTISSP 200. Our recommendation is that the present structure continue until a new administration can address revising NSDD-145. Our thinking is that the revision should concentrate on how the Law should be implemented in the Executive Branch.

III. 2. Mr. Walker: How does the Public Law affect the scope and authority of the NTISSC/SAISS structure?

Mr. Gallagher: The Public Law has, in effect, clarified the scope and authority of the NTISSC by assigning the responsibility for leadership in protecting unclassified sensitive information to NIST.

III. 3. Mr. Walker: What is the effect of the Public Law on NTISS issuances?

Mr. Gallagher: The main emphasis of the Public Law is on the protection of computer systems handling unclassified sensitive information in the Federal Government. The NTISS issuances were provided for in NSDD-145 and were meant to cover both classified and nationally sensitive but unclassified data. Those voting on passage of the NTISS issuances came from a wide range of non-DoD Government agencies. Since the Public Law was passed, the directive portion of this policy as it applies to unclassified sensitive data in the Government is up in the air. We hope that the unclassified portion of the Government will agree that C2 is a minimum level of trust.

III. 4. Mr. Walker: What is the NIST position on this?

Dr. Katzke: Agencies that process unclassified sensitive information in computer systems that are accessed by more than one user, when those users do not have the same authorization to use all of the information processed or maintained on the computer system, should conduct a risk assessment, and based upon its results, should consider using operating systems providing Controlled Access Protection (i.e., C2 level of protection) to enhance user access controls. If available and cost effective, operating systems that have been evaluated as C2 by the NCSC should be used. If there are no such evaluated products available for use, then agencies can select operating systems that best meet the criteria of C2 until such products become available.

Agencies that process unclassified sensitive information in shared computer systems that need, as determined by a risk assessment procedure, a system-enforced (i.e., software/hardware) mandatory access control policy, should consider using operating systems with Trusted Computer Security Evaluation Criteria (TCSEC) Division B or A protection. If available and cost effective, operating systems that have been evaluated as B or higher by the NCSC should be used. If there are no such evaluated products available for use, then agencies can select operating systems that best meet the criteria of the desired level of protection until such products become available.

A mandatory access control policy is applicable to situations where there are clearly identified, separate categories of data within a computer system and users that are only authorized access to a subset of the categories. B-level operating systems provide system (i.e., software/hardware) enforcement of the mandatory policy. They preclude the ability of a user to use his/her discretion to pass data of one category to any user who is not authorized to receive information of that category.

NCSC-evaluated operating system products can provide a base upon which to implement integrity and availability controls. Agencies that process unclassified sensitive information needing protection against loss of integrity and availability should consider using evaluated products for this base, provided the evaluated products meet the agencies' requirements.

III. 5. Mr. Walker: How will NSA support this requirement for a minimum level of protection as controlled access protection (C2) by 1992 for the unclassified and classified communities? What are the practical implications for vendors and agencies? What will be done to ensure that adequate products meeting these requirements are available? Will the NCSC evaluation process fill this need?

Mr. Sohmer: We support the use of C2-level approved systems to protect sensitive unclassified information by 1992. Both the DoD and the Intelligence Communities have incorporated the NTISSP requirements into their policies and regulations. We are encouraging these communities to understand the C2-level protection features and apply them as needed in their systems. Note my emphasis on "as needed." We must apply common sense to the issuance. The assumption is that one of two things is important to the user: (1) separation of data is important on the system to be protected; and (2) sharing of data is a system feature.

I do want to emphasize "minimum" in your question--that is, NTISSP is defining a minimum level of trust at C2. I believe the real answers to many of our problems are at the higher levels of trust. We begin really to help in virus control, for example, at B2. Also, most of our assurances that the system is robust, from a security viewpoint, come at higher levels. Therefore, C2 was thought of as a first step. We believe that many organizations need MAC-based systems to help control the effects of Trojan horses and viruses. Also, if we really ever expect to process multi-levels of sensitive information on one computer system or over a network, we must move up to B-level systems. Meanwhile, we project 33 systems will be available at C2 or higher by 1990.

IV. TRUSTED SYSTEM TECHNOLOGY

Mr. Walker: The overall information security problem is considered by many to consist of three aspects: confidentiality of information, integrity of information, and assurance of service. In many systems confidentiality is the overriding concern and good theoretical foundations exist for confidentiality protection within the Orange Book and its interpretations. In some systems, integrity and assurance of service are more important than confidentiality. The integrity issue in all of its many aspects (e.g., correctness of data, accuracy of communications, correctness of systems operation) is more complex because the term is used to describe a broad collection of related requirements. Many of these requirements are far less understood and have weaker theoretical foundations than confidentiality. Within the trusted computer base, the mechanisms required to protect and enforce integrity controls are similar to those required to enforce confidentiality controls. Without strong integrity mechanisms, assurance of service cannot be guaranteed.

The Orange Book represents a sound technical basis in systems with more than one user for provision of confidentiality protection and some aspects of integrity (e.g., correctness of system operation) protection and should be employed in designing and evaluating systems requiring this protection. At present no equivalent criteria exist for the broader aspects of integrity or assurance of service.

Is this fair representation?

NSA (Mr. Gallagher): Yes

NIST (Mr. Burrows): Yes

IV. 1. NSA, through the NCSC, has a well-established program for the evaluation of industry trusted computer system products. Does NIST plan to establish a separate trusted computer system evaluation criteria or get into the product evaluation business?

Dr. Katzke: No. This is an area in which NIST intends to draw upon NSA. NIST intends to issue guidance on when and how to use the criteria, to extend the criteria to include provisions for integrity and availability, and to issue standards in this area when the criteria are mature and measurable.

IV. 2. Mr. Walker: Since the passage of the Public Law, have you observed any change in vendor interest in producing trusted products/technology?

Mr. Sohmer: Yes. The vendor interest has increased. More evaluations are being requested, and our workload is greater.

TRUSTED SYSTEM TECHNOLOGY

In my opinion, however, this increase has not been due to the Public Law. It comes from the momentum achieved under NSDD-145 and the many procurements coming out of the Government that reference levels of trust. The fact that there has not been a decrease in enthusiasm since the Public Law is evidence to me that the vendors really understand all the benefits--especially the marketing ones--of trust technology. I believe that in the context of data sharing and data separation in computer systems, vendors are convinced that trust technology is the "only game in town" providing a base upon which to build trusted applications.

IV. 3. Mr. Walker: In what specific areas of computer security does NIST plan to develop standards?

Dr. Branstad: NIST believes that several different types of standards are required in the computer security area. Interoperability standards are required in which one system, or user, wishes to transfer information to another system, or user. Interface standards are required in which one component of a system either connects to another component or replaces another component. Standards of good practice are needed to establish acceptable procedures. Definitional standards, such as glossaries, are needed simply to clarify terminology and to develop new languages for effective communication.

NIST believes that the following areas are subjects for standards in one or more of these types:

Operating System Security Interfaces (e.g., POSIX)

Network Security Protocols (e.g., OSI, ISDN)

Application Program Security (e.g., DBMS, EFT, MHS, EDI)

Physical Security (e.g., computer environments, security systems)

Logical Security (e.g., authentication, authorization, labels)

Personal Security (e.g., identification, personalization)

IV. 4. Mr. Walker: The C-level of the Orange Book provides what is called "discretionary security" while the B-level and above provide "mandatory security." A B-level system is "better" in a trusted system sense than a C-level system. Please describe the advantages of mandatory access control over discretionary access control.

Mr. Sohmer: They are mechanisms that enforce different policy. Mandatory access control, based upon protected labels, is an inherently stronger mechanism than discretionary control; MAC

TRUSTED SYSTEM TECHNOLOGY

provides greater guarantees that the policy is enforced. With DAC the owner of a file grants file access to others at his discretion. However, once that file has been copied by the person to whom access has been granted, the original owner has lost the ability to influence who may subsequently see and alter his copy of the data. In the case of MAC, the TCB-controlled labels guarantee that the organization's security policy is enforced regardless of how the current owner of the information may attempt to share it (e.g., medical records will be accessible only by those with the proper accesses). The system "mandates" this. Going one step deeper, it's well known that, from the viewpoint of confidentiality and integrity of data, DAC is vulnerable to Trojan horses and viruses and thus DAC, to some extent, relies upon the goodwill of the system users. MAC, on the other hand, provides much greater assurance that the intended policy is indeed enforced. An effective MAC implementation protects files against Trojan horses or viruses. Overall, one can say that DAC enforces the desires of the holder of the data, while MAC enforces organizational policy.

IV. 5. Mr. Walker: At present, there are relatively few products on the Evaluated Products List while there seems to be much evaluation activity under way, and we keep hearing about a long queue of vendors waiting to get into evaluation. What is the status of products on the EPL, in evaluation, and in the queue for evaluation? What are your plans, if any, for accelerating this process and for dealing with the reevaluation of upgraded products?

Mr. Sohmer: By the end of 1990, we expect to have an Evaluated Products List consisting of 71 products, 33 of which are complete systems, all of which are C2 or higher. That ought to solve the problem of a scarcely populated EPL. We are also looking toward research efforts to help us evaluate B1 and below products faster. The formal portion of the evaluation currently takes about six months for a C2 product. We would like to shorten this time using expert systems and their tools.

The RAMP (Rating Maintenance Phase) of the computer security product evaluation program provides for the maintenance of computer security ratings across product revisions. The purpose is to provide current versions of trusted products to the public. RAMP is essential for this purpose because of the frequency with which many vendors revise their product offerings. Vendors frequently offer new releases of a product every few months and keep multiple versions under development at all times. Without having the benefit of RAMP, only the formally evaluated version is a "trusted system" with a current Orange Book rating. RAMP allows the NCSC to establish a rating and an EPL listing for each product release which may follow after a formally evaluated product release. RAMP will normally yield an EPL listing for a revised product within a few weeks of

its release date. RAMP builds upon a formal product evaluation and does not provide an opportunity to avoid a formal evaluation. We will require the vendors to participate in the RAMP program for each of their formally evaluated products. This will become part of our Memorandum of Agreement with them. Essentially the RAMP position changes the Orange Book requirement for configuration management and introduces it at C2 (vice B2).

IV. 6. Mr. Walker: Many RFPs are asking for products meeting various levels of the criteria and are stating that while NCSC evaluation is useful, it is not required. How do you feel about this situation? Isn't it undercutting the efforts of the NCSC by allowing vendors to claim a level of evaluation by "emphatic assertion?"

Mr. Sohmer: I don't view this as a threat at all or an "undercutting." It puts the role of the accreditor and the rater into perspective. The accreditor assesses his risk and signs up to it using evidence at hand about the data, the users of the system and the environment. It is proper for an accreditor to tell vendors what his agency needs and decide who meets those needs. I'd much rather have a user say, "I need an A1," than know he needs an A1 but not put that in his RFP because there's nothing appropriate yet on the EPL. We are in the early stages of laying out a methodology along these lines as guidance for users--i.e., how to specify their needs on RFPs--aka a procurement guideline. The ultimate solution is a well-populated EPL. However, until we get to this point, the accreditor must assess the risk and decide what products meet his needs regardless of whether they are on the EPL or not. We are also studying ways of informing users just how far into the process of evaluation a product has progressed. This will help the decision-maker as he decides which system is best for his needs.

IV. 7. Mr. Walker: Is there anything the NIST can do to help this situation?

Dr. Katzke: NIST recommends that evaluated products be chosen over unevaluated products whenever available and cost effective.

IV. 8A. Mr. Walker: The Orange Book is technical criteria for evaluating computer systems and is most useful to computer scientists in the vendor and industry system development community and to Government evaluators. Users of computer systems typically aren't computer scientists but need help determining what levels of trusted systems are needed for their applications. The NCSC "Yellow Book" is a simplistic approach to an

TRUSTED SYSTEM TECHNOLOGY

environmental guideline, equating trust requirements with the range between the highest level of classified data and the lowest cleared user. The Navy's environmental guideline attempts to introduce factors of system architecture into the determination of level of trust required. What is the status of efforts to further understanding in this vital area for the national security community?

Mr. Sohmer: The Yellow Book has done an excellent job in this area because of its simplicity. However, one problem is that a guideline such as this tends to be viewed as gospel rather than guidance. The Yellow Book translated the Orange Book Classes to specific DoD environments and was more subjective than most of our publications. We encourage the wise translation of the Orange Book classes into other environments. There ought to be a Yellow Book for various parts of the civil sector. Also, someone in a segment of the private sector would be wise to translate to his environment. Then it would be interesting to analyze which levels of trust are needed in the various constituencies. Another help would be a procurement guideline that really works. We're trying to put out such a document in the next year. Also, we need to publish more "design guidance" so that the users will understand how to build security into their environment by using the building blocks of trust technology.

IV. 8B. Mr. Walker: Do you agree with Mr. Sohmer's opinion on the need for an equivalent approach for the Government sector that deals with unclassified sensitive data (i.e., a Yellow Book for each constituent)?

Dr. Katzke: NIST has requested the assistance of the NCSC in developing guidance for users of trusted products in sensitive unclassified data security applications.

IV. 9.A. Mr. Walker: Has any requirements analysis been done to determine the need for trusted products in the unclassified segments of the Federal Government?

Mr. Sohmer: Last year we tried to determine how well trusted products were selling in the segments of the Federal Government that process sensitive unclassified data. Our analysis showed that the need for trusted products is easily recognized for varying levels of trust throughout the Federal agencies. Because of the lack of data directly from the agencies, our analysis was based on several assumptions. We hope that the security plans being written and submitted by the Federal departments and agencies will provide insights into the requirements for trusted products for this constituency. Trust technology is a relatively new concept that will take off as people and agencies learn more about it. The fact that all the major vendors are incorporating at least C2 levels of trust into

TRUSTED SYSTEM TECHNOLOGY

their product lines will help because agencies will begin to use these features whether they originally thought they needed them or not.

IV. 9 B. Mr. Walker: Has there been any analysis of requirements for trusted products for classified systems?

Mr. Sohmer: In 1984, our survey within DoD showed the requirements for trusted products for classified systems, in accordance with the best available information. Presently that information is four years old and is of limited value because it does not cover requirements for classified systems outside of DoD.

V. CRYPTOGRAPHY

- V. 1. Mr. Walker: Please describe the recent history of developments regarding the Data Encryption Standard algorithm and NIST's plans regarding DES.

Dr. Branstad: NIST conducted the second five-year review of DES in 1987 during which 34 comments were received. The review was to help decide whether the DES would be reaffirmed, withdrawn, or revised. As a result of the overwhelming recommendation to reaffirm the DES without revision, the Secretary of Commerce reaffirmed the DES as the only standard to be operationally used for protection of unclassified information in the Federal Government.

For the next five years, NIST will continue to validate hardware implementations of the DES and add them to the list of DES devices that may be procured by Federal organizations. NIST has identified an ongoing need for effective, efficient, low-cost publicly known cryptographic algorithms and associated standards for a wide variety of unclassified security applications. While DES satisfies the need in most of these applications, other algorithms and standards may be needed in the future.

- V. 2. Mr. Walker: What are NSA's plans regarding DES?

Mr. Rainville: NSA ceased accepting DES products for Federal Standard 1027 endorsement on 1 January 1988. We currently have three data products in our evaluation pipeline and will complete these evaluations. At the time the decision was made, it was our intention to address new applications with CCEP equipment so as not to have all our eggs in the DES basket.

NSA has supported and will continue to support acquisition and keying requirement for endorsement products.

- V. 3. Mr. Walker: NSA has dropped its program to endorse FS 1027 products. What are NIST plans in this area?

Dr. Branstad: Federal Standard 1027 was published by the General Services Administration and covered the physical and logical security of telecommunications security devices implementing the DES. NSA, which developed the specifications of the standard, also endorsed products meeting the standard until the end of 1987. Through recent legislation, NIST was given the responsibility for telecommunications standards in addition to information processing standards. NIST plans to revise the specifications that were contained in FS 1027, review the new specification through the normal public review process, and issue a new Federal Information Processing Standard (number

CRYPTOGRAPHY

140) on this topic. Meanwhile, NIST plans to announce a waiver procedure to FS 1027 in the interim.

- V. 4. Mr. Walker: NSA has been emphasizing in their Commercial COMSEC Endorsement Program (CCEP) Type I and Type II cryptographic systems. Would you describe what Type I and Type II systems are, the status of these activities, and how they relate to DES?

Mr. Rainville: CCEP Type I products are industry-produced, highly secure systems intended for protection of classified information. Type II products are intended to provide security for unclassified but sensitive government information. Type II products were viewed as the successors to DES. Both passage of Public Law 100-235 and the response from the marketplace have caused us to reexamine and refocus our CCEP Type II plans. We have not yet concluded this reexamination.

- V. 5. Mr. Walker: Any NIST follow up comments?

Dr. Branstad: NIST believes that a number of algorithms are required to satisfy a range of requirements. Type I algorithms satisfy the need for classified application requirements. Type II algorithms satisfy the requirements for protecting data in special Government applications within the United States that require levels of protection beyond those required for commercial, financial, or personal privacy applications. A third family of algorithms and security products is needed to satisfy the requirements for protection of unclassified data in these latter application. These algorithms must be implemented in low-cost products that can be produced and used throughout the U.S. and also exported for applications requiring protected data communications between foreign locations and locations within the U.S. The DES satisfies some of the requirements associated with Type II algorithms and many of the requirements associated with the third family of algorithms, sometimes called Type III. Type I and II algorithms are defined and controlled by NSA. A Type III family of algorithms is being discussed by NIST and NSA. We have requested the assistance of NSA in defining this family and obtaining algorithms that would satisfy the requirements that have been identified by NIST.

Mr. Rainville: NSA is discussing with NIST areas in which we will provide cryptographic support and those will be identified in the MOU.

- V. 6. Mr. Walker: What is the Secure Data Network System (SDNS) and what impact will it have on network security evolution?

CRYPTOGRAPHY

Mr. Gallagher: SDNS is the Secure Data Network System. This project considers how to implement network security in a data network, with the addition of some of the latest security technologies and products, while continuing to provide the same level of service on the network. You will be hearing more about these data security concepts and how the vendors will be building network security products for retrofit into existing network technologies. The principal computer vendors in the U.S. have been involved in the development of these security specifications. NSA has released the security protocol specifications at layer 3 and five others are in review for release.

Like the Future Secure Voice project, SDNS is aimed at bringing low-cost security products and easy-to-use key management to the networking technologies of the 1990s. It is and will continue to be multi-vendor oriented and centered on international communication standards such as the OSI suite. The key management component is now being developed on contract. Several CCEP and one funded effort are in development. Given that the OSI suite of protocols becomes widely used, SDNS specifications will speed up the implementation of security services in the evolution of network security.

V. 7. Mr. Walker: What are NIST's plans for using SDNS technology?

Dr. Branstad: NIST and NSA have worked together to develop a single security architecture that would include security services, protocols, and mechanisms that satisfy the data communication protection requirements of both classified and unclassified information. The SDNS project is based on the OSI architecture that is now being supported by many computer vendors as a part of their commercial product offerings.

NIST intends to use the results of the SDNS project in developing OSI security standards if and when the results become publicly available. We have already included references to SDNS in Federal Information Processing Standard 146 (GOSIP) that requires Federal agency procurements of computer networks to solicit OSI products. We are working with NSA in making the results of the SDNS project publicly available so that all computer vendors have an equal opportunity to utilize them. NSA has released two of the documents that have resulted from the SDNS project and NIST intends to review the protocols (SP3 and SP4) as the basis for FIPS. We would like to develop a comprehensive family of standards for OSI security and will use the other protocols and key management provisions of the SDNS project when available. If they are not made public, NIST plans to use or develop other standards in order to have a complete security system for OSI networks.

CRYPTOGRAPHY

- V. 8. Mr. Walker: What is the relationship between SDNS and the Orange Book or TNI? What trust characteristics or levels are required to satisfy SDNS requirements?

Mr. Sohmer: The SDNS and Orange Book or TNI technologies complement each other, but should be viewed independently. The SDNS specifications are aimed at allowing security services to be provided to computer networks predominantly through cryptographic techniques. With respect to trust characteristics, depending upon application requirements, the computers and network security components may have any of the levels of trust discussed in the Orange Book and TNI. SDNS requirements alone do not dictate a specific trust rating. However, it makes little sense to provide strong cryptography and low assurance. Therefore, we visualize that the lowest level of trust would be C2 for system high environments.

- V. 9. Mr. Walker: Any NIST comments?

r. Branstad: We believe that network security is of utmost importance in processing sensitive and valuable data. Security must be provided at two generic places in a distributed network: within end systems and between end systems. The OSI architecture includes all communication services between end systems but does not include the data processing services within end systems. OSI security similarly protects data during communication but not during processing or storage in the end system. SDNS provides a subset of OSI security services and therefore is necessary but not sufficient for a trusted network.

We support use of the SDNS security protocols that have been developed for electronic mail (layer 7) and between end systems (layer 4). Putting cryptographic security within an end system requires a significant level of trust of the end system. The end systems must be trusted to separate users, users' data, users' processes, users, communications, and the cryptographic keys that may be associated with each user, process, communication, or end system. An integrated security system is thus dependent on both trust technology and cryptographic technology and we are working towards this integration.

We support use of the SDNS security protocol at intermediate systems (layer 3), which are often called gateways. The granularity of protection is less than if implemented within end systems, but fewer security systems are required. Trust technology is needed within the intermediate systems where security is implemented.

- V. 10. Mr. Walker: Does NIST have any plans to develop public key cryptographic-based security systems?

CRYPTOGRAPHY

Mr. Burrows: NIST has been requested by a number of agencies and organizations to develop public key cryptography standards for access authorization certificate exchange and for digital signatures. NIST is considering the development of such standards and has requested NSA's assistance in these efforts. PL 100-235 requires NIST to develop standards for protecting sensitive unclassified information in all Federal computer systems. Computer systems are defined to include those that switch, interchange, transmit, or receive data or information. Since cryptographic-based systems are the only known way of protecting data between geographically distributed computer systems, NIST believes that cryptographic-based security (including integrity, confidentiality, and authentication) standards must be issued for computer networks and distributed systems. We intend to pursue satisfying the need for cost-effective security systems through the standards process and intend to work with NSA in this area.

V. 11. Mr. Walker: Any NSA comments?

Mr. Gallagher: We will work with NIST in this area.

VI. INFOSEC PRODUCTS

VI. 1. Mr. Walker: In late 1985, NSA reorganized its traditional COMSEC and newly formed computer security activities into an Information Security organization with a new office designed to merge the two technologies and evaluation processes to promote production of integrated products. What is the status of this integration and what are plans for future activities?

Mr. Gallagher: You are correct, we did establish a new office at NSA to bring together these two disciplines with the goal of providing INFOSEC products. We did not merge the evaluation processes, however, of COMPUSEC and COMSEC. There has been some talk of eventually doing so, but I don't think that will happen any time soon. Concerning the new office - they are working on several products that you may have heard about and will hear about at this conference. Some were discussed in detail yesterday at Overture Day. Some of these projects are BLACKER, SDNS, LEAD, and GILLAROO.

VI. 1A. Mr. Walker: What is your advice to a vendor who wishes to build an INFOSEC product and have it evaluated by NSA? How long will the evaluation take?

Mr. Gallagher: Everyone knows that this is currently a long process. The problem comes when one intersects COMPUSEC and COMSEC techniques. Because there are different issues involved, each with a different analysis tradition, deadlocks occur in determining how trusted a given INFOSEC device is. The two evaluation groups at NSA are working together on how to streamline the process. We are working jointly on two evaluations and hope to streamline the process. Our advice is to contact NSA's Industrial Relations Office to request an evaluation.

VI. 2. Mr. Walker: What is your advice to a vendor that wishes to merge computer and telecommunications security technologies?

Dr. Branstad: NIST supports the merging of computer and telecommunications security technologies. We believe that cryptography can be used effectively within a computer system to segregate users and data as well as assure data integrity, control access, and authenticate users. We believe that computer security technology is required in communication switches and gateways in which either data is unencrypted or the encryption/decryption processes are being performed. We do not

INFOSEC PRODUCTS

intend to separate standards into one application or the other because they cannot be separated in most instances.

VII. TRUSTED NETWORK INTERPRETATION/
TRUSTED DATABASE INTERPRETATION

VII. 1. Mr. Walker: It has been over a year since the NCSC published the TNI. What feedback has been received and what are NSA's plans for the TNI?

Mr. Sohmer: We have received voluminous feedback on the TNI. We welcome such cooperation and feedback. This information helps us to know where to focus our attention, as well as which vendors and users need our help. One of our pursuits will be Appendix A, "Evaluation of Network Components" section, which addresses network components that still cannot be evaluated or assigned ratings. We find the real issues haven't been laid out for such network components. Those that can be evaluated do not fit naturally into the policy structure of the TNI. We need to define the basic issues and work on how to evaluate these components.

In Part II, "Other Security Services" section of the TNI, the criteria are too subjective. These criteria are based on analysis and testing, in a subjective context without specifying the hows and how much; therefore, these criteria are difficult to use. We have more work to do in this area.

The NCSC is working toward putting out a new version of the TNI in the Fall 1989. Additionally, supplemental guidance is being written. A networks environment guideline is currently in early draft form. This will provide guidance on appropriate levels of network systems and network components for specific environments, i.e., a Yellow Book for network environments. A trusted network testing guideline is also in draft form. This document is intended to tell vendors what is expected of them in the way of test plans, procedures, and documentation for a product that is to be evaluated.

We also spent about six hours yesterday at Overture Day going into details of where we are relative to the TNI.

VII. 2. Mr. Walker: What are NIST plans for utilizing the TNI?

Dr. Katzke: The TNI is a product of the NCSC and thus will be used as appropriate in the network security standards and guidelines developed by NIST. NIST participated in the workshop on computer network security that led to the TNI. The document contains a great deal of information about trusted networks, OSI security, and network security services and mechanisms. We intend to use this and additional information being incorporated in Version 2 of the TNI where appropriate in our network security standards activities. We understand that more

objective confidentiality and integrity criteria are to be included in Version 2, and that more use is to be made of the Security Architecture addendum to the Open Systems Interconnection Protocol Reference model. NIST concurs in these goals and looks forward to reviewing Version 2.

VII. 3. Mr. Walker: What is the status of the pending Trusted Database Interpretation?

Mr. Sohmer: We are proceeding toward the release of a first draft of the Trusted DBMS Interpretation. We mentioned last year that a draft would be available by now, but some of the difficult issues contained in this document have taken longer to resolve than we originally anticipated. The TDI is now in first draft form and will be distributed to a wide group of experts. We intend to receive their comments and integrate them to produce a second draft. The schedule calls for release of the final TDI in September 1989. As the TDI has been developed, we have discovered that it could be generalized into an interpretation for trusted applications. We are discussing that option.

VII. 4. Mr. Walker: Any NIST comment on the TDI?

Mr. Burrows Security of data bases and data base management systems is of fundamental importance in commercial data processing systems. We have concentrated our efforts on the development of data base management standards and have not addressed the security issues to date. We look forward to reviewing the TDI.

VIII. TRUSTED UNIX ACTIVITIES

VIII. 1. Mr. Walker: Both NIST and NCSC appear to be involved with supporting/sponsoring different standard-setting activities on UNIX, specifically POSIX and TRUSIX. What are the goals of these activities and are these competing activities?

Mr. Sohmer: The TRUSIX group was formed by the NCSC in June of 1987 to provide guidance for developing trusted UNIX based-systems. TRUSIX was formed to help vendors by resolving the important security issues that they would need to face when developing their own trusted UNIX-based systems, and to help evaluators face the anticipated influx of trusted UNIX products by giving them something to compare purported trusted UNIX-based systems to. After identifying Orange Book class B3 as the target, TRUSIX set out to develop a descriptive top-level specification (DTLS) for UNIX that would serve as a "roadmap" for vendors developing trusted UNIX-based systems. We chose B3 as a target because it contained all of the feature requirements of the Orange Book, and we believed that the guidance from TRUSIX would trickle down and, in effect, define the issues for the lower classes as well. Another output of the TRUSIX effort is a companion document that will serve as a guide to implementors of the specification. This companion document will take a detailed look at the Orange Book features, for example, MAC, DAC and audit and will discuss cautions and concerns about implementation options, including generic architecture issues.

POSIX defined its scope at the June P1003.6 meeting and the two groups are working towards a common view. That common view is a TCB interface specification based on the B3 requirements of the Orange Book. The way in which the two activities differ is that the output of POSIX will be a standard and because of this will be subject to a more intense development process. It may not be available for several years, whereas we plan to have the TRUSIX guidance available by mid-1989.

Dr. Katzke: NIST has made a strong commitment to the portable operating system environment, or POSIX, standards effort in general and to the development of appropriate security interface specifications as part of the evolving family of POSIX standards. NIST is working actively with the IEEE in the standards development effort with the objective of adopting the IEEE P1003 standards as Federal Information Processing Standards (FIPS). We expect that all future Federal procurements of systems with requirements for application portability will be required to conform to the POSIX FIPS. To ensure that security features were part of the POSIX standards, NIST sponsored the formation of the P1003.6 Security Working Group and provides the chairman of that committee.

TRUSTED UNIX ACTIVITIES

The IEEE P1003.6 and the NCSC TRUSIX efforts are and need to be complementary efforts and should not be allowed to become "competing" activities. It would be a great disservice to the Government and industry if we were to develop conflicting "standards" and thereby put users in the position of buying either portability or security because, frankly, they would more than likely opt for the former.

The scope of the IEEE effort is to develop specifications for standard interfaces to security features or functionality in POSIX-conforming implementations. It is important to recognize that the scope of the IEEE effort cannot and does not attempt to address underlying implementations (and therefore cannot really address assurance issues). It is the intention of the P1003.6 effort to include specifications for security features found in the Orange Book -- and these will include Access Control Lists (a B3 feature). But again, the POSIX security specifications will be for interfaces only, and will not address the internal, architectural considerations. The real objective is to provide security functionality for the wide range of user communities without precluding the features needed for Orange Book evaluation.

The role of the TRUSIX effort, as we see it, is to provide guidance to vendors developing UNIX-like operating systems for Orange Book evaluation. This clearly must address assurance and even architectural issues in addition to system interfaces. Those system interfaces, however, will need to conform to those specified in the POSIX standards if vendors wish to sell to the Federal Government. Thus, it will be extremely important that the NCSC/TRUSIX activity continue to be represented in the P1003.6 effort.

IX. INTERNATIONAL SECURITY STANDARDS,
EXPORT CONTROLS

IX. 1. Mr. Walker: What is the NIST position relative to international security standards?

Mr. Burrows: We have long encouraged and supported the adoption of U.S. industry standards as Federal standards and as international standards wherever possible. Such standards allow U.S. vendors to compete easily in international markets, and the cost of such products is generally lower because of a large market. Within the constraints of economic and national security interests in each country, we support the same policy with respect to security standards. However, our first priority is the protection of sensitive data within the Federal Government and then within the U.S. Our next priority is the protection of information that is transmitted internationally but which may affect the economic foundation of the U.S.

IX. 2. Mr. Walker: What is NSA's position relative to international security standards?

Mr. Rainville: NSA supports appropriate international security standards, especially those which promote worldwide interoperability. In addition, NSA recognizes NIST's experience in the international standards process and believes that together we can produce and establish appropriate international security standards.

IX. 3. Mr. Walker: There are legitimate concerns about the transfer of sensitive technology to foreign organizations, both foe and friend. Nevertheless the very nature of international business causes even fewer computer manufacturers to be able to show a pedigree that contains no foreign ownership or involvement. What is NSA's present position on evaluations of products of U.S. companies with a significant level of foreign ownership (or of products of non-U.S. companies)?

Mr. Gallagher: The present NSA policy is to perform evaluations with companies who have been judged acceptable according to their foreign ownership, control, and influence. Any conflict due to foreign ownership, control, or influence must be resolved in a manner acceptable to NSA, such as a voting trust or proxy arrangement. This overall policy is being reviewed by NSA management at the present time. This review will address the initial evaluation, as well as the effect of a change in ownership or control after the initial evaluation has been completed.

INTERNATIONAL SECURITY STANDARDS, EXPORT CONTROL

Dr. Branstad: NIST has performed validations for DES devices and for devices implementing ANSI X9.9 and X9.17 for any organization, both foreign and domestic, that has requested a validation. NIST does not differentiate between foreign and domestic companies when performing conformance tests.

IX. 4. Mr. Walker: What is the U.S. Government's position on export control of trusted products?

Mr. Rainville: Export of these products is controlled under the International Traffic in Arms Regulations (ITAR). Also, currently, Orange Book systems at the B3 and A1 levels must be licensed. A license is currently required for export of DES-based encryption products. Licenses for individual sales or distribution of products may be obtained. NSA generally recommends approval of licenses for DES-based products if:

1. the end user is a bona fide financial institution or
2. a U.S. entity is to use the product for protection of its own information or
3. the product is used for access control applications.

We do not anticipate any export approvals for CCEP equipment.

IX. 5. Mr. Walker: What is the U.S. Government's position of export control of cryptographic products?

Mr. Rainville: Type I and II COMSEC products cannot be exported without a license and it is unlikely that licenses will be granted. Export of DES products is still subject to the International Traffic in Arms Regulation (ITAR). Applications for licenses for DES-based products are usually approved for: (1) financial end uses, such as EFT protection; (2) access control application, e.g., PIN or password protection; (3) use by a U.S. company, regardless of application, if used to protect its own information.

SECTION 2: QUESTIONS SUBMITTED BY SESSION ATTENDEES

- 2.1. Public Law 100-235 requires agencies to develop a security plan for "sensitive" systems. NIST and NSA indicate they will jointly review these plans. Guidance issued at the 8 July workshop was not very clear as to the results of the review. What is the purpose of the plans? To determine adequate protections are in place? To define weaknesses of protection within Government? To police enforcement of the law?

JOINT ANSWER: Our understanding of Congress's purpose in requiring plans is to ensure that planning is done and that feedback is obtained from a source knowledgeable in security issues. The Law states that NIST and NSA will review these plans. Neither NIST nor NSA has been given any authority to "police enforcement of the law."

See OMB Bulletin No. 88-16. See answer to question II.6. for more details.

- 2.2. What procedures have been established to review security plans, which are due 8 Jan 89?

JOINT ANSWER: NIST and NSA will review the plans as a team effort, and will develop review criteria.

See answer to question II.6. for more details.

- 2.3. To comply with the Law, security plans must be submitted for review by Jan 1989. 1. Within what timeframe do NIST and NSA foresee these plans actually being implemented? 2. When will training for managers and users be provided? 3. When do you expect managers to incorporate security into their programs? 4. When do you anticipate that users will accept the potential constraints inherent in operating in a more secure system environment?

JOINT ANSWER to Question #1: The timeframe for implementing these plans will be determined by the agency.

JOINT ANSWER to Question #2: The timeframe for providing the training for managers and users will be determined by the agency.

JOINT ANSWER to Question #3: There is existing guidance from OMB requiring inclusion of security into the agency program. See OMB Circular A-130, Appendix III, Security of Federal Automated Information Systems.

JOINT ANSWER to Question #4: This will occur as the training programs are instituted, as further security awareness is

QUESTIONS SUBMITTED BY SESSION ATTENDEES

developed, and as the cost of not operating securely is recognized.

- 2.4. Was the Washington Post story that the NIST will have just a few minutes to review each of the security plans for each sensitive system true? Can you describe what you will be looking for in those plans? Does NIST have authority to reject those plans?

JOINT ANSWER: Both NIST and NSA are responsible under the Law to review these plans, and it may take up to one year. NIST and NSA will provide feedback, which in some cases will make clear that particular plans are not adequate. It will be the responsibility of those agencies receiving this feedback to take appropriate action. In all cases we will be looking for a common sense approach to computer security.

See answer to question 2.2., OMB Circular A-130, and OMB Bulletin 88-16 for more information.

- 2.5. Public Law 100-235 states that a security plan is or should be approximately ___ [sic] pages long. What about the computer center's plan? In order for my activity to comply with the OPNAVINC our security plan must be approximately 35 to 40 pages.

JOINT ANSWER: We expect the guidance issued by OMB to be satisfied. We believe that this can be accomplished in five pages or less.

See OMB Bulletin 88-16.

- 2.6. OMB requires all agencies with sensitive systems to report. The reporting burden for each system may exceed the burden of developing a security plan. Many systems have plans in accordance with existing agency direction. Why examine individual plans? Management data/reports are more efficient, reduce reporting burden, free up time to execute plans. Why report in an OMB-dictated format? Agency rules frequently are as good or better. The cost of transposing/converting existing plan to OMB format is appalling and uses up valuable COMPUSEC resources.

JOINT ANSWER: See answer to questions 2.1. through 2.5.

- 2.7. If a Navy activity has complied with the OPNAVINST 5239.1A by submitting their ADP Security Plan to COMNAVDAC and that plan has been approved, do they still have to submit their ADP Security Plan to NSA/NIST?

QUESTIONS SUBMITTED BY SESSION ATTENDEES

JOINT ANSWER: See answer to question 2.6.

2.8. Because NIST's personnel resources are very limited, from where are the reviewers of the system security plans going to come? If from within NSA, specifically where?

JOINT ANSWER: NIST and NSA are required to review these plans. We will do that as a team using resources from both organizations. See answer to question II.6. for more details.

2.9. How many people from NSA will be reviewing the security plans?

JOINT ANSWER: NIST and NSA are currently reviewing exactly how many resources from each organization will be devoted to this effort.

2.10. Will the security review team have prior training before they review the security plans?

JOINT ANSWER: Yes.

2.11. When will the security review team be formed?

JOINT ANSWER: We expected to form the team in late 1988.

2.12. Who are these 60,000 plans submitted to?

JOINT ANSWER: Agency submissions should be sent to:

Computer Security Review Team
National Institute of Standards and Technology
A216, Technology Building
Gaithersburg, MD 20899

2.13. Who is going to review them?

JOINT ANSWER: See answer to questions II.6, 2.8, and 2.9.

2.14. What are the criteria for review?

JOINT ANSWER: The review criteria is being developed.

2.15. Does the Public Law require separate security plans for civil agency computers which are approved by their DAA to process classified data?

JOINT ANSWER: Public Law 100-235 does not require that security plans be provided for systems that handle classified information.

QUESTIONS SUBMITTED BY SESSION ATTENDEES

If no, what about data integrity issues that may not have been covered by the approval to process classified (i.e., only concerned with disclosure)?

JOINT ANSWER: Data integrity questions, as they relate to computer security, should be addressed by your security strategy and approved by your DAA.

2.16 . How does the Computer Security Act of 1987 affect the Computer Fraud and Abuse Act of 1986?

JOINT ANSWER: In general there is no direct relationship between the two laws. The Computer Fraud and Abuse Act discusses the penalty for illegal access to a Federal computer. The Computer Security Act establishes NIST as the agency responsible for providing guidance to protect unclassified sensitive information in the Federal Government.

2.17. Who can offer guidance for prosecuting violations of the public laws in general and PL 100-235 in particular?

JOINT ANSWER: If you need guidance, we suggest that you consult a lawyer.

2.18. How would you interpret the application of NTISSP #200 and other directives to the protection of telecommunications systems which carry traffic for which all "users" do not have the same level of authorized access?

JOINT ANSWER: NTISSP #200 is not directly related to the protection of telecommunication systems. It sets policy requiring a minimal C2 level of trust by 1992 in computer systems where appropriate. Other directives provide more information about protecting various levels of traffic on telecommunications systems. Since there are various applicable directives, depending on the type and security classification of information being handled, and varied levels of authorized user access, all appropriate applicable guidance/directives must be considered and applied for each specific telecommunications system. See also the answer to questions III.4. and III.5.

2.19. The Armed Forces Communications & Electronics Association (AFCEA) working group's report to NSA on sensitive information in the private sector delineates about 28 explicit categories of non-DoD classified information that companies consider to be sensitive. How might these criteria be transmitted to Government agencies to assist them in identifying their areas of sensitivity and criticality.

JOINT ANSWER: In asking this question, you have helped make agencies aware of this report. Assuming the report is

QUESTIONS SUBMITTED BY SESSION ATTENDEES

unclassified, interested agencies should contact AFCEA about its availability.

- 2.20. Does NIST have any jurisdiction over or guidance for sensitive unclassified computers operated by the Department of Defense?

JOINT ANSWER: Yes. The exclusions for DoD are those covered by the Warner Amendment. These include ADP systems related to intelligence activities, cryptologic activities, command and control systems of the military forces, and weapon systems.

See Appendix B.

- 2.21. I am concerned about the use of the term "storage of sensitive information" in Public Law 100-235. The dictionary definition of "information" is the knowledge gained from data. Based on personnel experience, mind set, and the current circumstances, one could respond to the number "7" as days in the week, money refunded, or the number of people working at a particular facility who have AIDS. The point I raise is, if we can store data and not information, what compliance problems confront us--especially if incidents reach the courts?

JOINT ANSWER: We suggest that you contact a lawyer.

- 2.22. Do you think that the technology required to comply with the Public Law is available? If not, what plans do you have to: (a) identify requirements and areas where research is required; and (b) support research to provide the necessary technology?

JOINT ANSWER: There are many technical, management, and physical security controls already available to enable agencies to comply with the Computer Security Act. See answers to questions III.4, III.5, and V.1. See answers to questions IV.3, V.5, V.7, VII.2, and VIII.1 for discussions of future requirements.

- 2.23. How can we get on a mailing list for the meetings being held in connection with the Computer Security Act of 1987?

JOINT ANSWER: NIST maintains a mailing list for notification of meetings to be held in connection with the Computer Security Act. Requests to be included on this list may be sent to:

National Institute of Standards and Technology
Computer Security Division
A216 Technology Building
Gaithersburg, MD 20899
ATTN: PL100-235 Mailing List

QUESTIONS SUBMITTED BY SESSION ATTENDEES

- 2.24. How will the timetable for the requirements of the Computer Security Act of 1987 affect the computer/network vendors?

JOINT ANSWER: The ongoing security planning in each Federal agency will highlight security issues and undoubtedly focus attention on solutions. Many of the solutions needed will be in the network security area. This will require additional trusted products.

- 2.25. If NIST has the responsibility for assisting Federal agencies with computer security as prescribed by the Public Law, will NIST continue to charge agencies for any assistance provided and why? It seems that lately NIST will only help if you are willing to pay for their help.

NIST ANSWER: NIST provides general technical assistance and support through conferences, workshops, publications, briefings, and other activities that are available to participants Government-wide. Agency-specific projects involving long-term support and deliverables are carried out on a cost-reimbursable basis under agreements with sponsoring agencies.

- 2.26. What impact will the Computer Security Act have on security policies and procedures in the financial institution industry? Specifically, does "unclassified but sensitive" include items such as Federal fund transfers and ACH (Automated Clearing House) transactions? If so, will there be any requirements for data classification among smaller institutions?

NIST ANSWER: NIST has been working with the Department of Treasury and the Federal Reserve to develop standards for protecting electronic fund transfers. NIST is also supporting the development of voluntary industry standards for electronic fund transfers. Under the Computer Security Act, NIST will continue to transfer appropriate technology to other communities.

- 2.27. What impact will the Act have on state agencies, such as revenue and tax authorities which indirectly pass and capture information for the Federal government (IRS)?

JOINT ANSWER: The Computer Security Act of 1987 covers state computer systems that process sensitive Federal information.

- 2.28. For those activities within DoD that process multiple levels of automated sensitive information, both classified and unclassified, will we be required to use two sets of standards depending on which type job is being run? Is any coordination under way to ensure that the two standards are compatible?

QUESTIONS SUBMITTED BY SESSION ATTENDEES

JOINT ANSWER: If you mean that you are not operating in multi-level secure mode and your computer is processing both classified and unclassified data, then, by definition, you are operating a classified system. The rules for such operation within the DoD are clear.

2.29. Who has responsibility for defining requirements for protecting sensitive unclassified information that is directly related to classified contracts?

JOINT ANSWER: You should get guidance from the security officer assigned to the classified contract. If he determines that the information falls under the Warner Amendment, then NSA will provide guidance. Otherwise, NIST will.

2.30. What happens to NTISSP 200, i.e., "C2 by 1992"?

JOINT ANSWER: See answers to questions III.1, III.2, III.3, III.4, and III.5.

2.31. What are the NCSC and NIST going to do, if anything, to:
(1) achieve international harmony among standards/criteria?
(2) gain international agreement on respect of evaluation results?

JOINT ANSWER: See answers to questions IX.1 and IX.2.

2.32. Given the split in responsibilities between NIST and NCSC regarding sensitive unclassified and classified information, how will NIST migrate their guidance into the DoD and military departments?

JOINT ANSWER: See answer to question 2.30

2.33. How will NIST guidance be interweaved into other DoD guidance when the object system is one which processes sensitive and classified data (multilevel in nature)?

JOINT ANSWER: See answer to question 2.29.

2.34. Does this law affect DoD contractors? If so, how?

JOINT ANSWER: This law covers computer systems operated by Federal agency contractors. Security plans for those systems should be included in the plans submitted by the Federal agency.

2.35. Given a system that contains unclassified entries that are classified in the aggregate, who has authority over the system and who has the authority to determine the classification of the aggregate?

JOINT ANSWER: The System Security Officer has the security responsibility for the system, and the security classification

QUESTIONS SUBMITTED BY SESSION ATTENDEES

officer provides guidance on the classification level of the aggregate information.

- 2.36. The Public Law exempts systems that process intelligence-related information. Many systems that process intelligence also process sensitive unclassified data. Are such systems still exempt?

JOINT ANSWER: If the sensitive unclassified information is related to the intelligence, it is exempt under the Warner Amendment.

- 2.37. How does this law affect the private sector?

JOINT ANSWER: PL 100-235 does not directly affect the private sector.

See Answer to question 2.26.

- 2.38. What effort will NIST make to accredit civilian systems, or will this task fall to NSA?

JOINT ANSWER: See answer to question IV.1.

- 2.39. How does NIST plan to adequately cover all their responsibilities when they have been quoted as saying they don't have the resources to look at even the computer virus problem? Does NIST plan to "raid" NSA for personnel?

NIST ANSWER: NIST will try to do the best job it can with resources provided by Congress.

- 2.40. NIST has received additional funds for developing security training/awareness programs and materials. Is anyone taking the lead in acquiring the necessary funds at a Government-wide level to pay for training classes for managers and users?

NIST ANSWER: Federal agencies are responsible under the Act for providing computer security awareness training. NIST is developing materials to be used for this training.

- 2.41. Where is NIST going to get the manpower/billets to carry out their mission?

NIST ANSWER: See answers to question II.6.

- 2.42. Could NIST comment on whether there should be a new Orange Book and environments guideline for the non-DoD sector?

JOINT ANSWER: See answers to questions IV.8A. and IV.8B.

QUESTIONS SUBMITTED BY SESSION ATTENDEES

- 2.43. Will NIST undertake a program to evaluate telecommunications products as appropriate for use in protecting unclassified sensitive information in Federal telecommunications systems? If so, what role will NSA play? If not, how will this void be filled?

NIST ANSWER: See answers to questions V.1., V.3., V.5., V.7., V.9., V.10., and VI.2.

- 2.44. Have I misinterpreted what has been said this morning or has the NIST stated: (1) they agree the Orange Book does not address the questions of integrity and assurance of service; (2) they have no plans to establish such criteria and will rely upon the NCSC evaluations even though they are Orange Book-based.

NIST ANSWER: See answers to questions III.4, IV.7, and IV.8B.

- 2.45. Will NIST establish standards regarding sensitive information against which products will be evaluated? Will NIST do evaluations of products (at any level)?

NIST ANSWER: See answers to questions IV.1. and IV.3.

- 2.46. Is it safe to assume that the NCSC and NIST will cooperate with each other and coordinate activities, evaluations, etc., to allow a vendor to develop a single trusted system that satisfies both military and civilian system requirements? Does a vendor need to pick the sector in which to make an offering?

JOINT ANSWER: See answer to question IV.1.

- 2.47. Will there be a separate set of standards for DoD secure systems and commercial systems that fall into NIST's "unclassified, but sensitive" domain? (That is, will Orange Book and TNI be adopted as standards?)

JOINT ANSWER: See Plenary Session "opening session" comments, and questions in Section IV, Trusted System Technology.

- 2.48. Is NSDD-145 being rewritten under Public Law 100-235 and, if so, when will the rewrite be completed?

JOINT ANSWER: See answer to question III.1.

- 2.49. What are the expected repercussions to agencies in responding to the requirements of the Computer Security Act of 1987?

JOINT ANSWER: The Act is an opportunity for agencies to analyze, plan, and document their security requirements and to improve security awareness.

QUESTIONS SUBMITTED BY SESSION ATTENDEES

- 2.50. Who is responsible for DES key management? NIST or NSA? Who sets the policy?

JOINT ANSWER: See answers to questions in Section V on cryptography.

- 2.51. One way to increase the availability of trusted systems from vendors is to develop a commercial (business) recognition of the need for such systems. How will NIST/NSA support and further the transfer of ideas and technologies to the non-Government arena?

JOINT ANSWER: NIST will have principal responsibility for transferring security technology to the private sector. As appropriate and as requested, the NCSC will discuss COMPUSEC issues with private sector entities after coordination with NIST.

- 2.52. Despite all the trusted products, Orange Book, Red Book, etc., responsibility for security must start with the user. Integrity and assurance for all users cannot happen if top management gives their passwords to designated staff. When passwords are shared, not only is the user's data not secure, anyone sending data to the user has no assurances about who will have access to the data being sent!

JOINT ANSWER: Amen!

- 2.53. Is performance taken into consideration when evaluating a product? The object reuse parameters, when turned on, take tremendous resources (i.e., time) from a system. How can we find out about performance when soliciting an evaluated product?

NSA ANSWER: The impact on system performance is not explicitly considered during an NCSC product evaluation. Performance does not affect the security rating of a product. However, serious performance impact is usually noted in a final evaluation report just as especially good performance is also noted. Potential sources of information are the vendor and user familiar with the product. We would like to note, however, that security need not adversely affect performance. Good computer security is good engineering, especially at the higher levels of trust. Most performance problems can be controlled by the security coordinator who decides when and how much security is "turned on" in a system.

- 2.54. There are a lot of products being developed and evaluated that have application to small systems. What, if anything, is being done to develop "trusted" systems at the B1 to B3 range in the large-scale category (e.g., IBM 3080/3090 with MVS or VM type operating systems)?

QUESTIONS SUBMITTED BY SESSION ATTENDEES

NSA ANSWER: Most major U.S. computer vendors are working with the NCSC to develop trusted products. The decision about what level of trust they develop belongs to the vendor alone. The strongest incentive to vendors to develop products at higher levels of trust is user demand. Users are encouraged to put specific requirements for trust into their RFPs. They must also not water down those requirements. If users need B3 protection, they should demand it whether or not B3 products are currently on the EPL.

2.55. What services/assistance does the NCSC now offer to civil agencies?

JOINT ANSWER: When civil agencies request services/assistance, they are usually referred to NIST. If NIST cannot provide the services/assistance, the NCSC with NIST coordination may provide services/assistance. Civil agency requests are considered on a case-by-case basis.

2.56. We are seeing a lot of effort by allied governments and treaty organizations to develop security evaluation criteria, certification criteria, and processes. In some cases, these criteria seem to be at variance with the Orange Book and the NCSC process, with a possible negative impact on the international market for trusted products. What are NCSC and NIST doing to attain international coordination of security criteria so as to minimize the fragmentation of the trusted product market while preserving individual countries' prerogatives to protect their own information?

JOINT ANSWER: Each allied government or treaty organization has its sovereign right to adopt its own guidelines, criteria, and standards. We try to share as much information as possible with them in hopes of obtaining compatible interoperable guidelines, criteria, and standards. We are working closely with our NATO allies to develop a single set of criteria. Some NATO nations have already developed their own versions of the criteria, and we will continue to work with these countries to try to assure compatibility between their publications and ours. This, however, is a difficult task because, as the question properly notes, these nations must preserve their prerogative to protect their internal information in their own way. Among the most powerful incentives to all nations to agree upon uniform criteria are the vendors themselves, whose demands for uniformity will carry significant weight with their respective governments.

Also, see answers to questions IX.1 and IX.2.

2.57. How will the NCSC make available a sufficient number and variety of C2 systems by 1992?

NSA ANSWER: See answer to question IV.5.

QUESTIONS SUBMITTED BY SESSION ATTENDEES

2.58. Given the Computer Security Act of 1987's allocation of responsibilities, is not the name of the National Computer Security Center now obsolete? Are there any plans to change it to, e.g., to National Classified Computer Security Center?

NSA ANSWER: No.

2.59. What makes you assume that "a Ph.D. in computer science" helps one understand what is going on in the Yellow Book overall or in computer security in general? The computer science curricula at most universities are almost devoid of computer security courses. What is being done to alter this situation?

NSA ANSWER: NSA has a university outreach program through which they are working with colleges and universities to get more computer security courses into their computer science curricula. Course development includes computer security courses for the computer science major as well as for information systems management students. NSA is working with the ACM and IEEE computer science curriculum development committees to accomplish this. Both the University of Maryland and Idaho State University gave presentations at the 11th National Computer Security Conference on their computer security programs.

2.60. Does the NIST feel that integrity can be built as an "add-on" to the Orange Book evaluation of a system?

NIST ANSWER: Integrity is a complex concept and is not well understood at the present time. NIST is working with experts in the private sector to investigate the concept and its relationship to data quality and the value of data. Until there is a better grasp of what integrity entails, NIST cannot say how it will mesh with the Orange Book.

2.61. What measures will be taken to ensure that two standards that vendors will be required to comply with will not be established?

JOINT ANSWER: See answer to question IV.1.

2.62. Are there plans for updating the Orange Book? If so, who will be responsible for making and reviewing those changes? Also, are there plans to modify the TNI and establish it as a DoD standard?

NSA ANSWER: NSA has no plans to change the Orange Book at present. However, NSA is working on integrity and service assurance issues, and will consider including more information on these issues when they are well defined. NSA plans to publish a draft TNI version II in Fall 1989. See answer to question VII.1 for status of TNI. The TNI has not been considered as a DoD standard.

QUESTIONS SUBMITTED BY SESSION ATTENDEES

2.63. What is the average transit time, for example, of a B2 system, between when a developer believes he has a finished, evaluatable product and when the product is evaluated, rated, and on the EPL?

NSA ANSWER: Presently, a B-level product takes on the average twelve to fifteen months for the formal phase of the evaluation. We cannot predict the length of time for the development phase because that phase is almost completely determined and controlled by the vendor. Publication of a final evaluation report follows after the rating is awarded.

2.64. Will copies of the RAMP paper be made available? It is not in the proceedings.

NSA ANSWER: Yes, the RAMP presentation is included in the package of post-conference papers. Also, NSA plans to have the RAMP documentation ready in the next several months. NSA currently offers a short course on the RAMP process for vendors.

2.65. What is the status of Type II products? Will 1027 or Type II apply for unclassified but sensitive data?

JOINT ANSWER: See answers to questions V.4. and V.5.

2.66. Will NIST produce cryptographic standards replacing the aging DES?

NIST ANSWER: See answer to question V.1., V.4., and V.5.

2.67. We understand that NIST has had input to and supports NSA's SDNS program and that NIST will provide a test bed for some of these products. What is your opinion of the acceptance and usage of this type system in civil agencies and private environments?

NIST ANSWER: See answer to questions V.7. and V.9.

2.68. The current division of responsibility has made it impossible for vendors not already having 1027 endorsed products to have DES products endorsed for use by Federal Government agencies. DES-based systems continue to be sold under 1027, and existing 1027 vendors continue to make improvements under their existing 1027 endorsement. How will this situation be resolved to enable non-1027 vendors to participate in this marketplace, i.e., how do we remove the current roadblock?

JOINT ANSWER: See answer to question V.3.

2.69. What is the future role of DES in: (1) the Government in general and NSA in particular? (2) financial industry?

QUESTIONS SUBMITTED BY SESSION ATTENDEES

(3) private sector in general? (4) the international community?

JOINT ANSWER: (1) See answers to questions V.1., V.2., and V.5.

(2) With the reaffirmation of the DES in the financial industry, it is expected to continue to be used by the banking community as a voluntary standard.

(3) We expect the DES to continue to be used in other private sector applications at the discretion of the user.

(4) In the international community, as in (2) and (3) above, where permitted by national governments and U.S. export regulations.

2.70. Will NIST develop or endorse a new public domain cryptographic algorithm to replace DES?

NIST ANSWER: See answers to questions V.1. and V.5.

2.71. Who has the cryptographic responsibility for digital signature algorithms? Is anyone defining a central registry for digital signatures?

JOINT ANSWER: NIST has responsibility for digital signature algorithms when used for systems other than those exempted under the Warner Amendment. A central registry for cryptographic algorithms is being discussed by the voluntary standards community. If needed, a registry for digital signature algorithms could be part of that registry. As before, NIST will request support from NSA on INFOSEC issues as appropriate.

2.72. Some countries prohibit encryption in private sector transmissions. Routine message authentication on every message may be more valuable than better encryption. Will you provide guidance on message authentication as well as cryptography?

JOINT ANSWER: Guidance on message authentication has been provided in Federal Information Processing Standard 113, Computer Data Authentication, and in voluntary industry standard ANSI X9.9, American National Standard for Financial Institution Message Authentication (Wholesale).

2.73. Is the Data Encryption Standard (DES) encryption algorithm approved for protecting all unclassified sensitive U.S. Government data (including DoD unclassified national security related information)? If not, how soon will NIST and NSA make an alternative encryption algorithm available?

QUESTIONS SUBMITTED BY SESSION ATTENDEES

JOINT ANSWER: See answers to questions V.1., V.2., V.4., and V.5.

2.74. Are there plans for encryption devices that will be available for use by foreign nationals, e.g., a Type III system that we understand was started and is now on hold?

JOINT ANSWER: NSA and NIST are discussing and will continue to discuss these issues. No firm plans have been made to date.

2.75. Encryption systems that require export licenses are not useful to international business. Is this problem going to be addressed?

JOINT ANSWER: While we agree that the need to get a license for encryption equipment is not always convenient, it is overstating the case to say that licensing requirements makes cryptography "not useful to international business." All countries require some form of license for export of cryptography, and many of our export controls are the result of international agreements. We are addressing ways of making the licensing process less onerous, but some sort of process will always exist because protection of cryptographic techniques is vital to national security interests.

See answer to question V.5.

2.76. Does NIST foresee development of classified cryptographic algorithms (possibly with NSA guidance) or will all NIST cryptography be publicly developed and openly released?

NIST ANSWER: See answer to question V.5.

2.77. What is the status of the process to determine which Government agencies will use Type II encryption devices (i.e., bullet proof)?

NSA ANSWER: This policy is still under consideration at NSA.

2.78. What is the status of TRUSIX? Who is participating in the TRUSIX meetings? What documents, etc., have they produced to date? What is the schedule of meetings and publications for the rest of 1988 and for 1989? Who is the point of contact?

NSA ANSWER: The TRUSIX group is presently defining an interface specification for trusted UNIX-based systems based on the B3 requirements of the Orange Book. The TRUSIX membership includes the NCSC, Infosystems Technology, Inc. (ITI), AT&T, Harris, Gemini, Sun, Gould, Mitre, and the Institute for Defense Analysis. The TRUSIX group's publication schedule is as follows:

- ACL Proposal Issues Paper Dec 88
- A Formal Security Policy Model Apr 89

QUESTIONS SUBMITTED BY SESSION ATTENDEES

- B2 DTLs Jun 89
- MAC Proposal/Issues Paper Jun 89
- B3 DTLs Nov 89
- Trusted Path Proposal/Issues Paper Nov 89
- I&A Proposal/Issues Paper Nov 89
- Administrator/TCB Interface Proposal/Issues Nov 89
- Architectural Issues Paper Nov 89

The point of contact for the TRUSIX group is Dr. Charles Testa of ITI. He can be reached on (301) 345-7800.

- 2.79. Because of the importance of computer security and the increasing number of PCs throughout our society, why not require PC licensing before one can purchase a computer (similar to ham radio licensing)?

JOINT ANSWER: The widespread use of personal computers makes this impractical. It is not clear what the purpose of licensing would be.

- 2.80. Is there an effort under way to coordinate the implementation activities resulting from the requirements of both Public Law 100-235 and NSDD 298, which established a National Operations Security (OPSEC) Program? In my opinion, the intent and planning applicative to these two national security-related requirements are complementary. The NSDD-298 White House Fact Sheet states in part: "The operations security process involves . . . identification of critical information, analysis of threats, analysis of vulnerabilities, assessment of risks, and application of countermeasures."

JOINT ANSWER: Implementation of the Public Law and NSDD-298 is the responsibility of each Federal department or agency, under the oversight of OMB. Because the two official documents complement each other, we believe there should be information exchange on their implementation in each department/agency.

- 2.81. Can UNIX vendors leverage their effort by porting a product from vendors such as AT&T, Addamax, and SecureWare?

NSA ANSWER: Yes, it would be advantageous to consider using a vendor's trusted UNIX software product on any vendor's hardware, if it is truly portable to that specific hardware. The problem is that portability is not a trivial issue.

PUBLIC LAW 100-235—JAN. 8, 1988

COMPUTER SECURITY ACT OF
1987

Public Law 100-235
100th Congress

An Act

Jan. 8, 1988
[H.R. 145]

To provide for a computer standards program within the National Bureau of Standards, to provide for Government-wide computer security, and to provide for the training in security matters of persons who are involved in the management, operation, and use of Federal computer systems, and for other purposes.

Be it enacted by the Senate and House of Representatives of the United States of America in Congress assembled,

Computer
Security Act of
1987.
Classified
information.
40 USC 759 note.
40 USC 759 note.

SECTION 1. SHORT TITLE.

This Act may be cited as the "Computer Security Act of 1987".

SEC. 2. PURPOSE.

(a) IN GENERAL.—The Congress declares that improving the security and privacy of sensitive information in Federal computer systems is in the public interest, and hereby creates a means for establishing minimum acceptable security practices for such systems, without limiting the scope of security measures already planned or in use.

(b) SPECIFIC PURPOSES.—The purposes of this Act are—

(1) by amending the Act of March 3, 1901, to assign to the National Bureau of Standards responsibility for developing standards and guidelines for Federal computer systems, including responsibility for developing standards and guidelines needed to assure the cost-effective security and privacy of sensitive information in Federal computer systems, drawing on the technical advice and assistance (including work products) of the National Security Agency, where appropriate;

(2) to provide for promulgation of such standards and guidelines by amending section 111(d) of the Federal Property and Administrative Services Act of 1949;

(3) to require establishment of security plans by all operators of Federal computer systems that contain sensitive information; and

(4) to require mandatory periodic training for all persons involved in management, use, or operation of Federal computer systems that contain sensitive information.

SEC. 3. ESTABLISHMENT OF COMPUTER STANDARDS PROGRAM.

The Act of March 3, 1901 (15 U.S.C. 271-278h), is amended—

15 USC 272.

(1) in section 2(f), by striking out "and" at the end of paragraph (18), by striking out the period at the end of paragraph (19) and inserting in lieu thereof: "; and", and by inserting after such paragraph the following:

"(20) the study of computer systems (as that term is defined in section 20(d) of this Act) and their use to control machinery and processes.";

15 USC 278h.

(2) by redesignating section 20 as section 22, and by inserting after section 19 the following new sections:

15 USC 278g-3.

"Sec. 20. (a) The National Bureau of Standards shall—

“(1) have the mission of developing standards, guidelines, and associated methods and techniques for computer systems;

“(2) except as described in paragraph (3) of this subsection (relating to security standards), develop uniform standards and guidelines for Federal computer systems, except those systems excluded by section 2315 of title 10, United States Code, or section 3502(2) of title 44, United States Code;

“(3) have responsibility within the Federal Government for developing technical, management, physical, and administrative standards and guidelines for the cost-effective security and privacy of sensitive information in Federal computer systems except—

“(A) those systems excluded by section 2315 of title 10, United States Code, or section 3502(2) of title 44, United States Code; and

“(B) those systems which are protected at all times by procedures established for information which has been specifically authorized under criteria established by an Executive order or an Act of Congress to be kept secret in the interest of national defense or foreign policy,

the primary purpose of which standards and guidelines shall be to control loss and unauthorized modification or disclosure of sensitive information in such systems and to prevent computer-related fraud and misuse;

“(4) submit standards and guidelines developed pursuant to paragraphs (2) and (3) of this subsection, along with recommendations as to the extent to which these should be made compulsory and binding, to the Secretary of Commerce for promulgation under section 111(d) of the Federal Property and Administrative Services Act of 1949;

“(5) develop guidelines for use by operators of Federal computer systems that contain sensitive information in training their employees in security awareness and accepted security practice, as required by section 5 of the Computer Security Act of 1987; and

“(6) develop validation procedures for, and evaluate the effectiveness of, standards and guidelines developed pursuant to paragraphs (1), (2), and (3) of this subsection through research and liaison with other government and private agencies.

“(b) In fulfilling subsection (a) of this section, the National Bureau of Standards is authorized—

“(1) to assist the private sector, upon request, in using and applying the results of the programs and activities under this section;

“(2) to make recommendations, as appropriate, to the Administrator of General Services on policies and regulations proposed pursuant to section 111(d) of the Federal Property and Administrative Services Act of 1949;

“(3) as requested, to provide to operators of Federal computer systems technical assistance in implementing the standards and guidelines promulgated pursuant to section 111(d) of the Federal Property and Administrative Services Act of 1949;

“(4) to assist, as appropriate, the Office of Personnel Management in developing regulations pertaining to training, as required by section 5 of the Computer Security Act of 1987;

“(5) to perform research and to conduct studies, as needed, to determine the nature and extent of the vulnerabilities of, and to

Regulations.

devise techniques for the cost-effective security and privacy of sensitive information in Federal computer systems; and

"(6) to coordinate closely with other agencies and offices (including, but not limited to, the Departments of Defense and Energy, the National Security Agency, the General Accounting Office, the Office of Technology Assessment, and the Office of Management and Budget)—

"(A) to assure maximum use of all existing and planned programs, materials, studies, and reports relating to computer systems security and privacy, in order to avoid unnecessary and costly duplication of effort; and

"(B) to assure, to the maximum extent feasible, that standards developed pursuant to subsection (a) (3) and (5) are consistent and compatible with standards and procedures developed for the protection of information in Federal computer systems which is authorized under criteria established by Executive order or an Act of Congress to be kept secret in the interest of national defense or foreign policy.

"(c) For the purposes of—

"(1) developing standards and guidelines for the protection of sensitive information in Federal computer systems under subsections (a)(1) and (a)(3), and

"(2) performing research and conducting studies under subsection (b)(5),

the National Bureau of Standards shall draw upon computer system technical security guidelines developed by the National Security Agency to the extent that the National Bureau of Standards determines that such guidelines are consistent with the requirements for protecting sensitive information in Federal computer systems.

"(d) As used in this section—

"(1) the term 'computer system'—

"(A) means any equipment or interconnected system or subsystems of equipment that is used in the automatic acquisition, storage, manipulation, management, movement, control, display, switching, interchange, transmission, or reception, of data or information; and

"(B) includes—

"(i) computers;

"(ii) ancillary equipment;

"(iii) software, firmware, and similar procedures;

"(iv) services, including support services; and

"(v) related resources as defined by regulations issued by the Administrator for General Services pursuant to section 111 of the Federal Property and Administrative Services Act of 1949;

"(2) the term 'Federal computer system'—

"(A) means a computer system operated by a Federal agency or by a contractor of a Federal agency or other organization that processes information (using a computer system) on behalf of the Federal Government to accomplish a Federal function; and

"(B) includes automatic data processing equipment as that term is defined in section 111(a)(2) of the Federal Property and Administrative Services Act of 1949;

"(3) the term 'operator of a Federal computer system' means a Federal agency, contractor of a Federal agency, or other organization that processes information using a computer

system on behalf of the Federal Government to accomplish a Federal function;

"(4) the term 'sensitive information' means any information, the loss, misuse, or unauthorized access to or modification of which could adversely affect the national interest or the conduct of Federal programs, or the privacy to which individuals are entitled under section 552a of title 5, United States Code (the Privacy Act), but which has not been specifically authorized under criteria established by an Executive order or an Act of Congress to be kept secret in the interest of national defense or foreign policy; and

"(5) the term 'Federal agency' has the meaning given such term by section 3(b) of the Federal Property and Administrative Services Act of 1949.

15 USC 278g-4.

"SEC. 21. (a) There is hereby established a Computer System Security and Privacy Advisory Board within the Department of Commerce. The Secretary of Commerce shall appoint the chairman of the Board. The Board shall be composed of twelve additional members appointed by the Secretary of Commerce as follows:

"(1) four members from outside the Federal Government who are eminent in the computer or telecommunications industry, at least one of whom is representative of small or medium sized companies in such industries;

"(2) four members from outside the Federal Government who are eminent in the fields of computer or telecommunications technology, or related disciplines, but who are not employed by or representative of a producer of computer or telecommunications equipment; and

"(3) four members from the Federal Government who have computer systems management experience, including experience in computer systems security and privacy, at least one of whom shall be from the National Security Agency.

"(b) The duties of the Board shall be—

"(1) to identify emerging managerial, technical, administrative, and physical safeguard issues relative to computer systems security and privacy;

"(2) to advise the Bureau of Standards and the Secretary of Commerce on security and privacy issues pertaining to Federal computer systems; and

"(3) to report its findings to the Secretary of Commerce, the Director of the Office of Management and Budget, the Director of the National Security Agency, and the appropriate committees of the Congress.

Reports.

"(c) The term of office of each member of the Board shall be four years, except that—

"(1) of the initial members, three shall be appointed for terms of one year, three shall be appointed for terms of two years, three shall be appointed for terms of three years, and three shall be appointed for terms of four years; and

"(2) any member appointed to fill a vacancy in the Board shall serve for the remainder of the term for which his predecessor was appointed.

"(d) The Board shall not act in the absence of a quorum, which shall consist of seven members.

"(e) Members of the Board, other than full-time employees of the Federal Government, while attending meetings of such committees or while otherwise performing duties at the request of the Board

Chairman while away from their homes or a regular place of business, may be allowed travel expenses in accordance with subchapter I of chapter 57 of title 5, United States Code.

"(f) To provide the staff services necessary to assist the Board in carrying out its functions, the Board may utilize personnel from the National Bureau of Standards or any other agency of the Federal Government with the consent of the head of the agency.

"(g) As used in this section, the terms 'computer system' and 'Federal computer system' have the meanings given in section 20(d) of this Act."; and

(3) by adding at the end thereof the following new section:

"Sec. 23. This Act may be cited as the National Bureau of Standards Act."

National Bureau
of Standards Act.
15 USC 271 note.

SEC. 4. AMENDMENT TO BROOKS ACT.

Section 111(d) of the Federal Property and Administrative Services Act of 1949 (40 U.S.C. 759(d)) is amended to read as follows:

"(d)(1) The Secretary of Commerce shall, on the basis of standards and guidelines developed by the National Bureau of Standards pursuant to section 20(a) (2) and (3) of the National Bureau of Standards Act, promulgate standards and guidelines pertaining to Federal computer systems, making such standards compulsory and binding to the extent to which the Secretary determines necessary to improve the efficiency of operation or security and privacy of Federal computer systems. The President may disapprove or modify such standards and guidelines if he determines such action to be in the public interest. The President's authority to disapprove or modify such standards and guidelines may not be delegated. Notice of such disapproval or modification shall be submitted promptly to the Committee on Government Operations of the House of Representatives and the Committee on Governmental Affairs of the Senate and shall be published promptly in the Federal Register. Upon receiving notice of such disapproval or modification, the Secretary of Commerce shall immediately rescind or modify such standards or guidelines as directed by the President.

President of U.S.

Federal
Register,
publication.

"(2) The head of a Federal agency may employ standards for the cost-effective security and privacy of sensitive information in a Federal computer system within or under the supervision of that agency that are more stringent than the standards promulgated by the Secretary of Commerce, if such standards contain, at a minimum, the provisions of those applicable standards made compulsory and binding by the Secretary of Commerce.

"(3) The standards determined to be compulsory and binding may be waived by the Secretary of Commerce in writing upon a determination that compliance would adversely affect the accomplishment of the mission of an operator of a Federal computer system, or cause a major adverse financial impact on the operator which is not offset by Government-wide savings. The Secretary may delegate to the head of one or more Federal agencies authority to waive such standards to the extent to which the Secretary determines such action to be necessary and desirable to allow for timely and effective implementation of Federal computer systems standards. The head of such agency may redelegate such authority only to a senior official designated pursuant to section 3506(b) of title 44, United States Code. Notice of each such waiver and delegation shall be transmitted promptly to the Committee on Government Operations of the House of Representatives and the Committee on Governmental

Federal
Register,
publication.

Affairs of the Senate and shall be published promptly in the Federal Register.

"(4) The Administrator shall revise the Federal information resources management regulations (41 CFR ch. 201) to be consistent with the standards and guidelines promulgated by the Secretary of Commerce under this subsection.

Regulations.

"(5) As used in this subsection, the terms 'Federal computer system' and 'operator of a Federal computer system' have the meanings given in section 20(d) of the National Bureau of Standards Act."

SEC. 5. FEDERAL COMPUTER SYSTEM SECURITY TRAINING.

40 USC 759 note.

(a) **IN GENERAL.**—Each Federal agency shall provide for the mandatory periodic training in computer security awareness and accepted computer security practice of all employees who are involved with the management, use, or operation of each Federal computer system within or under the supervision of that agency. Such training shall be—

(1) provided in accordance with the guidelines developed pursuant to section 20(a)(5) of the National Bureau of Standards Act (as added by section 3 of this Act), and in accordance with the regulations issued under subsection (c) of this section for Federal civilian employees; or

(2) provided by an alternative training program approved by the head of that agency on the basis of a determination that the alternative training program is at least as effective in accomplishing the objectives of such guidelines and regulations.

(b) **TRAINING OBJECTIVES.**—Training under this section shall be started within 60 days after the issuance of the regulations described in subsection (c). Such training shall be designed—

(1) to enhance employees' awareness of the threats to and vulnerability of computer systems; and

(2) to encourage the use of improved computer security practices.

(c) **REGULATIONS.**—Within six months after the date of the enactment of this Act, the Director of the Office of Personnel Management shall issue regulations prescribing the procedures and scope of the training to be provided Federal civilian employees under subsection (a) and the manner in which such training is to be carried out.

SEC. 6. ADDITIONAL RESPONSIBILITIES FOR COMPUTER SYSTEMS SECURITY AND PRIVACY.

40 USC 759 note.

(a) **IDENTIFICATION OF SYSTEMS THAT CONTAIN SENSITIVE INFORMATION.**—Within 6 months after the date of enactment of this Act, each Federal agency shall identify each Federal computer system, and system under development, which is within or under the supervision of that agency and which contains sensitive information.

(b) **SECURITY PLAN.**—Within one year after the date of enactment of this Act, each such agency shall, consistent with the standards, guidelines, policies, and regulations prescribed pursuant to section 111(d) of the Federal Property and Administrative Services Act of 1949, establish a plan for the security and privacy of each Federal computer system identified by that agency pursuant to subsection (a) that is commensurate with the risk and magnitude of the harm resulting from the loss, misuse, or unauthorized access to or modification of the information contained in such system. Copies of each such plan shall be transmitted to the National Bureau of Standards

and the National Security Agency for advice and comment. A summary of such plan shall be included in the agency's five-year plan required by section 3505 of title 44, United States Code. Such plan shall be subject to disapproval by the Director of the Office of Management and Budget. Such plan shall be revised annually as necessary.

40 USC 759 note. **SEC. 7. DEFINITIONS.**

As used in this Act, the terms "computer system", "Federal computer system", "operator of a Federal computer system", "sensitive information", and "Federal agency" have the meanings given in section 20(d) of the National Bureau of Standards Act (as added by section 3 of this Act).

40 USC 759 note. **SEC. 8. RULES OF CONSTRUCTION OF ACT.**

Nothing in this Act, or in any amendment made by this Act, shall be construed—

(1) to constitute authority to withhold information sought pursuant to section 552 of title 5, United States Code; or

Public
information.

(2) to authorize any Federal agency to limit, restrict, regulate, or control the collection, maintenance, disclosure, use, transfer, or sale of any information (regardless of the medium in which the information may be maintained) that is—

(A) privately-owned information;

(B) disclosable under section 552 of title 5, United States Code, or other law requiring or authorizing the public disclosure of information; or

(C) public domain information.

Approved January 8, 1988.

LEGISLATIVE HISTORY—H.R. 145:

HOUSE REPORTS: No. 100-153, Pt. 1 (Comm. on Science, Space, and Technology) and Pt. 2 (Comm. on Government Operations).

CONGRESSIONAL RECORD, Vol. 133 (1987):

June 22, considered and passed House.

Dec. 21, considered and passed Senate.

10 U.S.C.A. § 2315

Warner Amendment

10 § 2315. Law inapplicable to the procurement of automatic data processing equipment and services for certain defense purposes

(a) Section 111 of the Federal Property and Administrative Services Act of 1949 (40 U.S.C. 795¹) is not applicable to the procurement by the Department of Defense of automatic data processing equipment or services if the function, operation, or use of the equipment or services--

(1) involves intelligence activities;

(2) involves cryptographic activities related to national security;

(3) involves the command and control of military forces;

(4) involves equipment that is an integral part of a weapon or weapons system; or

(5) subject to subsection (b), is critical to the direct fulfillment of military or intelligence missions.

(b) Subsection (a)(5) does not include procurement of automatic data processing equipment or services to be used for routine administrative and business applications (including payroll, finance, logistics, and personnel management applications).

Added Pub.L. 97-86, Title IX § 908(a)(1), Dec. 1, 1981, 95 Stat. 1117.

¹So in original. Reference to "(40 U.S.C. 759)" was probably intended.

**NATIONAL COMPUTER SECURITY CENTER EVALUATED PRODUCTS LIST
CERTIFICATES**

1. Product: Citadel Version 4.0 (Subsystem)
Company: Computer Security Coporation
Recipients: Dan Pfister, Computer Security Coporation,
Angel Rivera, Computer Security Corporation

2. Product: MPE V/E (C2 Rating)
Company: Hewlett Packard
Recipients: Carl Smolka, Hewlett Packard,
Jim Schindler, Hewlett Packard,
Ken Jordan, Hewlett Packard,
Dennis Lee, Hewlett Packard,
Andy Dooley, Hewlett Packard

3. Product: Private Access (Subsystem)
Company: Computer Accessories Corp.
Recipient: Tim Wickiser Computer Accessories Corp.

4. Product: MVS/XA with RACF (C2 Rating)
Company: IBM Corp.
Recipients: Christopher Arnold IBM,
Larry Wills IBM,
Bill Vance IBM,

5. Product: PRIMOS Version 21.0.1 DoD2a (C2 Rating)
Company: Prime Computer, Inc.
Recipient: Allan Dossett, Prime Computer, Inc.

6. Product: Cortana Personal Computer Security System Version
1.21 (Subsystem)
Company: Cortana Systems Corporation
Recipient: John Morris, Cortana Systems Corporation

7. Product: VAX.VMS Version 4.3 with Version 4 Security
Update (Rating - RAMP-C2)
Company: Digital Equipment Corporation
Recipients: Steve Woodard, Digital Equipment Corporation,
Ed Suffern, Digital Equipment Corporation

8. Product: IDX-50 (Subsystem)
Company: Identix Inc.
Recipient: Linda Rolando, Identix Inc.

**NATIONAL COMPUTER SECURITY CENTER EVALUATED PRODUCTS LIST
CERTIFICATES**

9. Product: X-LOCK 50 (Subsystem)
Company: Infosafe Corporation,
Recipient: Alfred Jorgensen, Infosafe Corporation

10. Product: DIALBACK Version 1.5 (Subsystem)
Company: Clyde Digital Systems Inc.
Recipients: Jim Murdakes, Clyde Digital Systems, Inc.,
Robert Clyde, Clyde Digital Systems, Inc.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY CERTIFICATES

1. Company: Codercard, Inc.
Company Representative: Paul Johnson
2. Company: Digitech Telecommunications, Inc.
Company Representative: Deepak Gulati
3. Company: Infomax Securities
Company Representative: David Howard
4. Company: Jones Futurex, Inc.
Company Representative: Don Thompson
5. Company: Federal Reserve Bank of Cleveland
Company Representative: David Stahl
6. Company: Communications Systems
Company Representative: Sandy Epstein ACS
7. Company: The Chase Manhattan Bank, N.A.
Company Representative: Seymour Sherman
8. Company: Racal-Quardata Limited
Company Representative: Robert DiNatale

OUTSTANDING PAPER AWARDS

Expert Systems in Intrusion Detection: A Case Study

Michael M. Sebring, Eric W. Shellhouse, Mary E. Hanna,
National Computer Security Center; R. Alan Whitehurst, SRI
International

*Implementing the Clark/Wilson Integrity Policy Using Current
Technology*

William Shockley, Gemini Computers, Inc.

Automated Audit Trail Analysis and Intrusion Detection: A Survey

Teresa F. Lunt, SRI International