UCRL-JC-124361

CONF-9606221--8

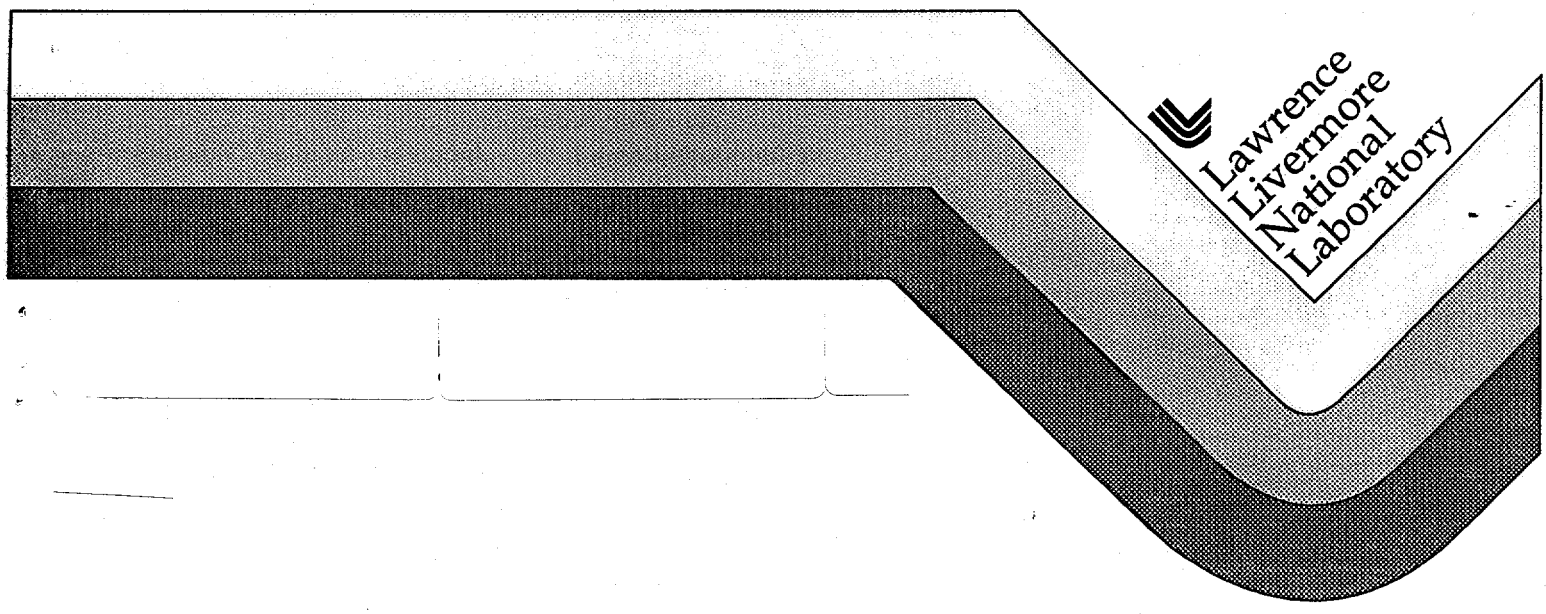
# Evaluating and Managing the Integrity of Computerized Accountability Systems Against Insider Threats

Edwin Jones
Alan Sicherman



This paper was prepared for submittal to the
American Defense Preparedness Association's
12th Annual Security Technology Symposium & Exhibition
Williamsburg, Virginia
June 17-20, 1996

June 1996



# EVALUATING AND MANAGING THE INTEGRITY OF COMPUTERIZED ACCOUNTABILITY SYSTEMS AGAINST INSIDER THREATS

Edwin Jones and Alan Sicherman, Fission Energy and Systems Safety Program
Lawrence Livermore National Laboratory, P.O. Box 808, L-634, Livermore, California 94551
(510)-422-8259

## INTRODUCTION

Accountability applications such as nuclear material accountability systems at Department of Energy (DOE) facilities provide for i) tracking inventories, ii) documenting transactions, iii) issuing periodic reports, and iv) assisting in the detection of unauthorized system access and data falsification. Insider threats against the system represent the potential to degrade the integrity with which these functions are addressed (e.g., altering data to misrepresent the quantity or location of nuclear material). While preventing unauthorized access by external threats is vital, it is also critical to understand how application features affect the ability of insiders to impact the integrity of data in the system. Aspects of modern computing systems including powerful personal computers and applications on networks, mixed security environments, and more users with increased software knowledge and skills help heighten the concern about insider threats.

Major benefits of computerized accountability systems result from empowering users to perform their jobs more effectively. However, potential insider "adversaries" with particular knowledge, skills and ability (KSA) can have increased opportunity to exploit system features. Relevant KSA areas include application software, database manipulation, electronics components, systems programming, and network communications[1]. The ability (and frequent facility desire) to modify software and *customize* functions and features such as user interface formats, access controls, and specialized statistical analyses or report generation provide even more possibilities for insider exploitation. On the other hand, a potential advantage of computerized applications lies in the various security features that they can incorporate. The way different aspects of software applications are customized for specific facilities can significantly impact the security effectiveness of the applications against the insider threat. A key question then arises as follows. How can managers and policy makers logically and pragmatically analyze the myriad customization and design options for accountability applications with respect to the insider threat?

In this paper, we describe a methodology for addressing this question, along with insights from its application to local area network (LAN) material accountability systems. The methodology was developed under the sponsorship of DOE's Office of Safeguards and Security in recognition of the need for more systematic, risk-based evaluations of nuclear materials accountability systems including those running on stand-alone mainframes, networks or client-server systems[2]. A key methodology goal in analyzing nuclear material accountability (MA) applications is to identify ways of preventing or mitigating possible additional risks from the insider threat as cost-effectively as possible, while assuring the integrity of the MA systems.

The methodology comprises a detailed yet practical taxonomy for characterizing diverse types of accountability system/software features and their implementation options. This taxonomy facilitates the systematic collection and organization of key information that helps spotlight such things as stages of information flow, transaction procedures or auditing procedures potentially susceptible to insider falsification. The methodology assists managers by providing them with a rational approach for identifying appropriate sets of protection features based on tradeoffs among operations concerns and the relative strengths and weaknesses of alternative controls in assuring the integrity of accountability systems.

## BASIC METHODOLOGY PARADIGM

An accountability system can be viewed as having subsystems or collections of functions/tasks. Major subsystems include inventories, material transactions, report generation, software maintenance, and network/system

traffic. Under each of these subsystems, we can develop lists of specific tasks or functions that are performed (e.g., such as *modify* accounting information for an item, under Material Transactions).

Associated with any task, a system may provide controls (or safeguards) to varying degrees including:

- Access/authorization controls
- Automated controls (for screening information submitted to the system)
- Human oversight controls (e.g., prior to officially updating a nuclear material database)
- Auditing/tracing controls (e.g., after database update)

Figure 1 schematically shows the relationship of these safeguards controls to MA insider threat concerns.

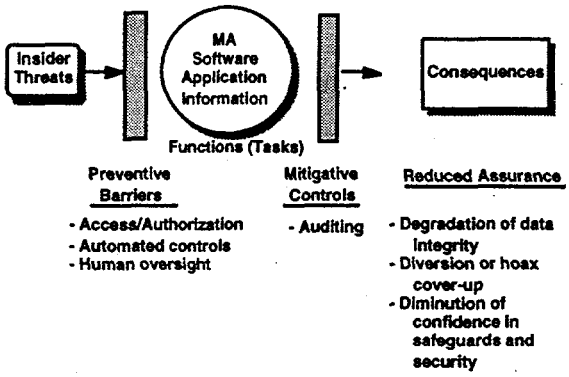



**Figure 1. Schematic of MA safeguards control categories and their relationship to insider threat concerns.**

We have developed *criteria* for analyzing the potential insider threat for each task of each subsystem in terms of the safeguards categories enumerated above. In the analysis approach, each criterion under the various control categories has an associated set of explicitly described possible *standard* gradations (or levels) tailored to each major subsystem. The gradations within each set are arranged from least preferable (weaker control) to most preferable (stronger control) from an insider threat perspective.

We have also developed a scheme for gathering information about applications in the form of subsystem and task *templates* for reflecting how each subsystem task is implemented. Based on the information provided on how a task is implemented, a correspondence or mapping is established between the task and one of the *standard* criteria gradations for each of the access, automated, human oversight and auditing criteria.

## Criteria for Evaluating Subsystem Tasks

We have formulated six criteria for evaluating application subsystem tasks vis-à-vis the insider threat. The first three criteria correspond to each of the three preventive barrier controls depicted in Fig. 1. The second three criteria characterize the mitigative auditing controls depicted in Fig. 1.

*Access/authorization refers to preventing anyone not authorized from accessing a particular subsystem task.* Gradations for this criterion reflect control provided by security features such as the password scheme that is implemented. For an authorized user, *automated controls* refers to application software features that can screen user input for things like format and numerical logic and consistency prior to updating a database. *Human oversight* is another control mechanism for screening user input.

For the auditing criteria, scrutability reflects the effort and expertise required to find task information sought in auditing records. *Resolution* reflects the kind of task detail present in auditing records (e.g., "what, where, when, who") and the nature of ambiguities that might exist about a task even after an audit record is examined. *Responsiveness* reflects the circumstances that actually *trigger* an analysis of audit records. Table 1 illustrates abbreviated descriptions of the levels for the responsiveness criterion tailored to the Material Transactions subsystem. (The actual level descriptions are more expansive and also contain more examples.) The criteria descriptions are explicit and tailored to each subsystem to help analysts and decision makers choose the criterion level that best describes their task implementation. For a different subsystem, the criteria descriptions contain levels and examples more appropriate to *tasks for that subsystem.*

**Table 1. Audit responsiveness criterion levels tailored to Material Transactions subsystem.**

| |
|---|
| 1 - only material anomaly (e.g., from inventory) triggers analysis of data available for audit |
| 2 - anomaly and random checks trigger analysis |
| 3 - anomaly and systematic audit analysis of special sensitive transactions (e.g., those accessing special data fields) |
| 4 - anomaly and random checks and systematic audit analysis of special sensitive transactions |
| 5 - all task transactions are proactively analyzed and auditor is also supplemented by automated search and flagging of audit records for any suspicious task patterns or discrepancies |

**Subsystem Task Information Collection Templates**

With a subsystem task template, an analyst selects the criteria gradations (levels) that best match the implementation of an application design. The concept is illustrated in Fig. 2. Each subsystem template (five templates to span the subsystems enumerated earlier) has its own set of tasks to be analyzed. For example, the Inventories subsystem tasks can include: generation of inventory checklist, transmittal of checklist, item check, bulk material measurement, measurement device calibration, recording of inventory data, transmittal of inventory data, matching of data to *books*, and discrepancy resolution. Material transactions tasks can include: modify, split, combine, move within a Material Balance Area (MBA), move between MBAs, move in/out of facilities, material in process and write-off type transactions, container operations, and Tamper Indicating Device (TID) operations.

The tasks for each subsystem have enough in common so the same tailored criteria descriptions reasonably apply to all of them. However, not all tasks in a subsystem need have the same safeguards controls as shown in Fig. 2 for illustrative design options. Generally, the list of tasks for each subsystem template should be comprehensive enough to cover: the different types of functions performed that could have different safeguards in their implementation, and the distinct stages of information or task flow involved in the subsystem.

Although simplified, Fig. 2 illustrates how the methodology described here helps to systematically formalize safeguards strengths and weaknesses of different design options for different tasks, and also how different application implementations can involve tradeoffs among safeguards controls. For Design A, automated and human oversight controls are relatively de-emphasized, while auditing controls are stressed. Design B has stronger automated controls but has less responsive auditing capability. Finally, Design C stresses human oversight controls while de-emphasizing auditing responsiveness.

The methodology in trials to date has provided a diagnostic tool that helps analysts and decision makers recognize in explicit fashion different safeguards control implications of design options. Often planners can improve on individual task safeguards controls by adding features that are relatively easy to implement. However,

adding certain extra safeguards features could exact significant penalties in operations. Thus *tradeoffs* often need to be considered about what safeguards to emphasize as illustrated in Fig. 2.

| | Subsystem - | | | | (audit) | | |
|---|---|---|---|---|---|---|---|
| | Tasks | access | automated controls | human oversight | scrutability | resolution | responsiveness |
| Design A: | Modify | 3 | 2 | 1 | 5 | 5 | 5 |
| | Split | 3 | 2 | 1 | 5 | 5 | 5 |
| | Move between MBAs | 3 | 3 | 4 | 5 | 5 | 5 |
| Design B: | Modify | 4 | 3 | 1 | 4 | 5 | 2 |
| | Split | 4 | 3 | 1 | 4 | 5 | 2 |
| | Move between MBAs | 4 | 3 | 1 | 4 | 5 | 2 |
| Design C: | Modify | 2 | 2 | 5 | 4 | 5 | 1 |
| | Split | 2 | 2 | 4 | 4 | 5 | 1 |
| | Move between MBAs | 2 | 2 | 4 | 4 | 5 | 1 |

**Figure 2. Illustrative simplified scoring of subsystem tasks.**

### Aggregating Across Tasks and Subsystems

In order to compare alternatives as to their effectiveness against the insider threat, the following issues must be addressed:

- how to quantify the safeguards value of criterion levels
- how criterion levels impact safeguards effectiveness for tasks
- how task effectiveness impacts subsystem effectiveness
- how subsystem effectiveness impacts overall effectiveness

Probabilities are used to quantify effectiveness in vulnerability assessment or VA tools[3]. An alternative with tradeoffs resulting in a higher estimated probability of adversary defeat is preferred when costs and operational impacts are acceptable. In a VA, a particular target is identified, and the probability of adversary defeat is understood in the context of a specific adversary action sequence. For evaluating MA software applications, however, we may not know what precise information compromise will actually result in any harm to the facility. What is needed instead is a practical characterization of how computerized application features affect the *level of effort* or KSA required by insiders to misrepresent or misuse data. This contrasts with a VA approach which would require defining many specific kinds of data compromise as well as estimating their probabilities of occurrence given insider attempts. Thus, for accountability applications, other approaches whose description is beyond the scope of this paper (e.g., multiattribute utility/value preference function theory[4]) allow for greater flexibility in evaluation when probabilities of adversary defeat are impractical to estimate[2]. With any approach, the key point is how to logically compare the efficacy of improving on one criterion versus another, or improving on one task versus another.

## METHODOLOGY APPLICATION TO LAN SYSTEMS

Several DOE facilities are currently in the process of gradually implementing the Local Area Network Materials Accountability System (LANMAS) to address their accountability needs. LANMAS comprises a collection of core procedures and illustrative applications/forms designed at Los Alamos National Laboratory for implementation on Microsoft's Windows NT Server Operating System[5]. However, each particular facility customizes its own implementation of LANMAS by: supplementing the core procedures with facility specific procedures, providing numerous forms and applications beyond the illustrative set that accompanies LANMAS, and adding code to the two predeclared site-specific stored procedures in each LANMAS supplied procedure[6]. Figure 3 illustrates schematically how forms, applications, software, and procedures can be organized on a client-server system.

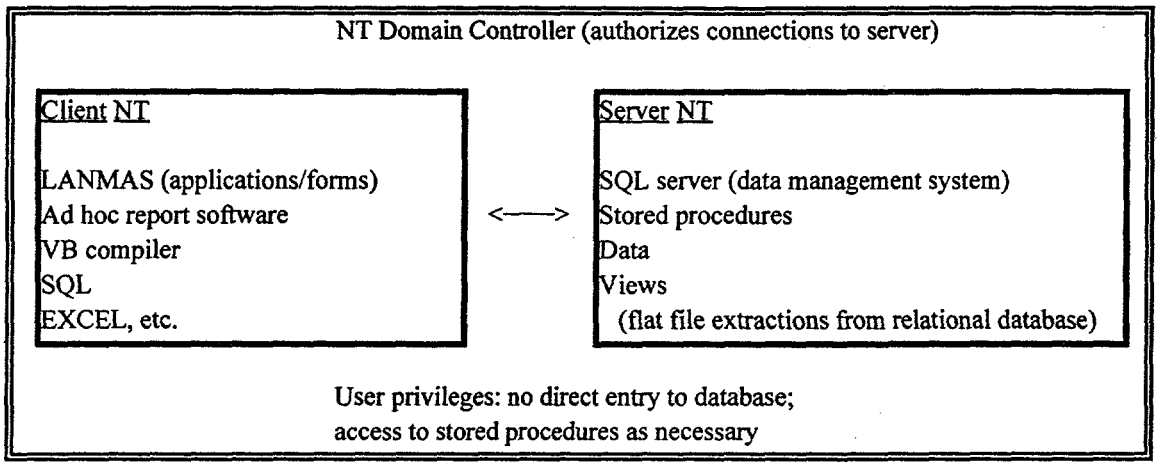



**Figure 3. Schematic of Windows NT client-server organization**

There are a variety of options that are possible when considering the types of safeguards that might be implemented for different tasks in such a client-server setup. Our methodology taxonomy helps planners systematically consider such possibilities for a LANMAS type system. We now discuss insights related to exploration of such task safeguards options for LANMAS.

### Preventive Barriers

**Access Controls.** With LAN systems and powerful client computers, an unprivileged insider may employ other techniques for obtaining passwords besides snooping or guessing. These include "sniffing" on the network line if passwords are sent unencrypted, or the use of a Trojan Horse (e.g., a program running on a client computer designed to mimic a system's standard logon procedure in order to read (and store away) sensitive data such as a user's password). Windows NT provides for encrypted password storage and controlled transmission, and the logon procedure in which users are trained involves a system boot sequence to counter Trojan Horses[7].

Data encryption in addition to password encryption is another safeguards feature that can be considered. There are gradations of encryption safeguards that can be applied to thwart a variety of network user attack scenarios including unauthorized: reading, changes (modifications), additions (replays) and deletions (filtering) of data[8,9]. Some techniques like crypto-checksums or message digests may be difficult to consider implementing on certain kinds of systems. There are both hardware and software options for implementing encryption. In certain classified network environments, approved hardware encryption devices are used to satisfy regulatory requirements.

A classified environment may sometimes provide additional safeguards features against insiders, even though the primary motivation for the environment may be to protect against outsiders. For example, special keys for the

encryption hardware in addition to passwords are required for using the system making access controls stronger. Sensitive printouts may be routed to secure vault rooms where they are handled by a two-person rule rather than having printout be accessible (and potentially susceptible to tampering or substitution) to more individuals. When the classified environment is no longer deemed needed, many of these safeguards may also disappear, even though they provide barriers to insiders. The methodology described here helps highlight the contributions of these kinds of safeguards controls against the insider threat, and allows decision makers to recognize when retaining some form of controls may still be desirable, even if the need for a classified environment should change.

**Automated Controls.** In a client-server system, there are a variety of possible schemes for implementing automated controls. These revolve around two issues: what controls should reside on the client versus the server, and what flexibility and capability is desirable for users on the client computers. There can be increased flexibility when stored procedures on the server can be called by any user created application (where the user is authorized to invoke the stored procedure), and where there are few restrictions on data input imposed by the stored procedures. An example that illustrates the need for flexibility in restrictions imposed by data checking involves the input of negative mass for materials involved in certain transactions. Some facilities need this flexibility in input while others may not. It may not be desirable to incorporate such a restriction in a core stored procedure.

LANMAS and custom facility applications and forms residing on the client computer can provide various kinds of checks on data entry before core procedures are called. However, such checks can be bypassed if users can directly invoke the stored procedure on the server. There can be mechanisms for a stored procedure to run only if it is invoked by an appropriate routine on a client. (When the actual call to the stored procedure occurs from a subroutine or function rather than being directly attached to the code processing an input field of a form, the subroutine or function is sometimes referred to as a *wrapper*.) Although providing an additional barrier, these too may be bypassed by tactics such as masquerading as the authorized routine, or by using codes that help disassemble the executable routines residing on a client which might then be modified and substituted. With a LANMAS allowing for significant facility customization, the actual source code for the routines residing on the client may also be readily available, as is the compiler (e.g., Visual Basic or VB) for the source code.

Additional automated control safeguards come from implementing the data checks on the server where software modification access is much more limited and privileged. A LANMAS example of this is where a facility enforces a *business rule* that material names within an MBA must be unique. To implement a check on material name user input data, the site specific stored procedure called at the beginning of the LANMAS core procedures for item movements, creating an item from bulk, receiving an off-site shipment and editing material names will do this check before continuing[6]. Because the check occurs inside the server stored procedure, it is more difficult to bypass.

Another flexibility versus automated control issue for client-server systems involves the desirability of using customized report software on client computers for individual user needs. Such software may utilize stored procedures to create tables from the database that are required, or may access *views* of the database already created on the server. While the data in the database itself is safeguarded by the server and data management controls, the reports are under the complete control of a user to manipulate once on the client computer. A facility can provide stored procedures that produce *official* accountability reports that do not involve client resident report generating software. However, it may not always be easy to make the distinction between the integrity and assurance of official reports and customized reports from a client computer. For example, the latter may be presumed to be as *reliable* as the former because the data is presumed to come from the same source. How report generating empowerment is implemented on client-server systems is an area worthy of cautious examination because of possible exploitation by insiders.

In summary with respect to automated controls, modern client-server systems can provide significant safeguards features for checking out data that can cause confusion before it is used to update the database. However, depending on how software for accomplishing tasks is implemented, the client-server setup can also create additional opportunities for insider data manipulation. We believe our methodology can help planners better recognize tradeoffs they are making between user flexibility and safeguards against insiders when choosing among implementation options.

**Human Oversight.** Most LANMAS tasks do not naturally make provisions for human oversight review of input (such as a two-person check) before a database is updated. However, some tasks like movement of material between MBAs involve first indicating a move to an *in transit* status from the sending MBA, and then a move from the *in transit* status to the receiving MBA. Each move is done by the separate custodian of the sending and

receiving MBA respectively. This is very much like providing input to a pending file which is done by one individual, but requiring a second individual's oversight before the data is transferred from the pending file. In this way, LANMAS can provide opportunities for human oversight controls prior to updating the status of material in an MBA.

### Mitigative Controls (Auditing)

**Scrutability.** The LANMAS philosophy involves preserving every single transaction record (without deleting any records) and uses a system of flags to indicate which records are active. Routines are provided for allowing a reviewer, for example, to trace an item's history from the records. While the actual file available for providing information for auditing analysis would be difficult to peruse sequentially for auditing information, a variety of querying and report routines can be assembled to help someone performing an audit to retrieve useful information with only modest effort.

**Resolution.** LANMAS preserves the important details of transactions. The actual computer on which a transaction was entered is not given much significance in the client-server setup where the user identification and privilege is the focus rather than the computer itself. (Even users with privileges to change software on the server may do so from a client computer.) Comments fields are available on forms for providing additional explanation and rationale for data entries besides the required input.

**Responsiveness.** Windows NT along with LANMAS can provide gradations of responsiveness depending on the implementation options selected. There may also be different gradations for different types of tasks and subsystems. For example, the audit information on material transactions may only be reviewed (i.e., an audit triggered) if a material anomaly occurs during an inventory. This degree of responsiveness is commonly observed at facilities. On the other hand, security logs which are kept by the NT system are often reviewed periodically (e.g., weekly) for certain activities whether or not a material anomaly has occurred. Security events that are audited can include such things as failed logon attempts and assignment of privileges. Windows NT can also send alert messages to designated individuals on security-related events[7].

LANMAS systems have the ability to provide more responsive *triggers* for recording information to be audited later such as when certain stored procedures are performed or certain database fields are accessed. One mechanism for doing this is via the site specific stored procedure described earlier which can check data input requests. Such triggers are often used in the debugging phase of implementing procedures and could be used for auditing purposes as well. If portions of LANMAS transaction files are to be reviewed periodically for selected types of activities, special routines will need to be written to collate out and organize pertinent information for the reviewer. Given the empowerment of users by some LANMAS implementations to perform tasks allowing flexibility of input at client computers with no additional human oversight, it can be critical to recognize when more proactive (responsive) auditing would be warranted to mitigate against insider threats.

## METHODOLOGY APPLICATION CONTEXTS

The methodology described here is intended to be flexible to meet the needs of information system planners and decision makers in general. The basic approach allows for easy modification or extension of details (such as the spectrum and gradations of criteria) to address a variety of contexts and decision making needs. Also, analysts can focus on one subsystem alone or use a subset of criteria in a modular analysis fashion.

In summary, we believe the methodology can help managers and policy makers:

- collect information about computerized applications relevant to the insider threat
- spotlight the relative strengths and weaknesses of application safeguards for a variety of accountability tasks using explicit criteria; the higher the criteria levels, the greater KSA required by an insider to misrepresent or misuse information
- evaluate tradeoffs between different system-software designs vis-à-vis effectiveness against the insider threat.

## ACKNOWLEDGMENT

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

## REFERENCES

1. D. S. Fortney and J. J. Lim, "A Technical Approach for Determining the Importance of Information in Computerized Alarm Systems," *Proceedings of the 17th National Computer Security Conference*, Baltimore, MD (October 1994).

2. E. D. Jones and A. Sicherman, "Analysis of Insider Threats Against Computerized Nuclear Materials Accountability Applications," *Proceedings of the 36th Annual Meeting of the Institute of Nuclear Materials Management*, Palm Desert, CA, Volume XXIV (July 1995).

3. R. A. Al-Ayat, C. J. Patenaude, T. A. Renis, and R. Saleh, "A Comprehensive Method for Evaluating Safeguards Against the Insider Threat," *Proceedings of the 30th Annual Meeting of the Institute of Nuclear Materials Management*, Volume XVIII (July 1989).

4. R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives*, New York: John Wiley & Sons, 1976.

5. *LANMAS Software Design Description, Revision: Draft-K*, Los Alamos National Laboratory, Los Alamos New Mexico, 4/29/96.

6. J. M. Davis, Jr., B. C. Osgood, S. M. Till, J. W. Wheeler, *Comprehensive Nuclear Materials Management System-Software Design Description for the Performance Based Incentive of 8/31/96, Draft: Revision A*, CNMMS_Project 96-05-24-0001, Savannah River Site, Aiken, South Carolina, 5/24/96

7. *Microsoft Windows NT 3.5 Guidelines for Security, Audit, and Control*, Redmond, Washington: Microsoft Press, 1994.

8. L. Mann, *Oracle Secure Network Services - Network Privacy and Integrity Technology*, Oracle Corporation, October, 1994.

9. J. I. Schiller, *Secure Distributed Computing*, Scientific American, November, 1994.