UCRL-JC-123327
PREPRINT

Assessing the Integrity of Local Area Network Materials Accountability Systems Against Insider Threats

E. Jones
A. Sicherman

This paper was prepared for submittal to the
37th Annual Meeting
Institute of Nuclear Materials Management
Naples, Florida
July 28-August 1, 1996

July 1996

Lawrence Livermore National Laboratory

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

ASSESSING THE INTEGRITY OF LOCAL AREA NETWORK MATERIALS ACCOUNTABILITY SYSTEMS AGAINST INSIDER THREATS

E. D. Jones and Alan Sicherman
Lawrence Livermore National Laboratory
Livermore, CA USA

ABSTRACT

DOE facilities rely increasingly on computerized systems to manage nuclear materials accountability data and to protect against diversion of nuclear materials or other malevolent acts (e.g., hoax due to falsified data) by insider threats. Aspects of modern computerized material accountability (MA) systems including powerful personal computers and applications on networks, mixed security environments, and more users with increased *knowledge, skills and abilities* help heighten the concern about insider threats to the integrity of the system. In this paper, we describe a methodology for assessing MA applications to help decision makers identify ways of and compare options for preventing or mitigating possible additional risks from the insider threat. We illustrate insights from applying the methodology to local area network materials accountability systems.

INTRODUCTION

Accountability applications such as nuclear material accountability (MA) 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. 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[1].

OVERVIEW OF ANALYSIS APPROACH

Taxonomy of Subsystems, Tasks and Criteria

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

Associated with any task, a system may provide controls (or safeguards) to varying degrees. We have developed *criteria* for analyzing the potential insider threat for each task of each subsystem in terms of safeguards controls. In the analysis approach, each criterion 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. The six criteria for evaluating application subsystem tasks vis-à-vis the insider threat are as follows.

Access/authorization (1) 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 (2)* 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 (3)* is another control mechanism for screening user input.

Additional auditing related criteria include the following. *Scrutability (4)* reflects the effort and expertise required to find task information sought in auditing records. *Resolution (5)* 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 (6)* 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.

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

1 - only material anomaly 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 or involving *negative mass*)
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

For a different subsystem, the criteria descriptions contain levels and examples more appropriate to *tasks for that subsystem*. The criteria descriptions are explicit and tailored to each subsystem to help analysts and decision makers choose the criterion level that best describes each task implementation.

We have also developed a scheme for gathering information about applications in the form of five subsystem task *templates* for reflecting how each subsystem task is implemented. Analysts record for each task the level for each of the access, automated, human oversight and auditing criteria that bests describes the task implementation. (See Reference 1 for more details of the approach and taxonomy.)

Aggregating Across Criteria, Tasks and Subsystems

To compare options, the following issues must be addressed:

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

Probabilities are used to quantify effectiveness in vulnerability assessment or VA tools. A VA approach would require defining many specific kinds of data compromise as well as estimating their probabilities of occurrence given insider attempts. For evaluating 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. Thus, for accountability applications, other approaches (e.g., multiattribute utility/value preference functions) allow for greater flexibility in evaluation when probabilities of adversary defeat are impractical to estimate. The key point is how to logically compare the safeguards efficacy of improving on one criterion versus another, or improving on one task versus another.

Multiattribute Utility Preference Models

Preference models based on multiattribute utility theory[2] have been developed to help decision-makers

logically and systematically address decision problems involving multiple attributes or criteria. These models adopt a set of reasonable preference assumptions that can be used to simplify the problem of comparing alternatives. With these assumptions, a decision maker (working with technical experts) indicates preferences for relatively simple decision situations to calibrate a preference model. The model is then used to establish how more complicated comparisons should be made to be consistent with the simpler comparisons and the preference assumptions.

More specifically, value judgment information is used to calibrate in a mathematically sound manner, a multiattribute utility function. Such a function takes as input for any alternative (e.g., a particular MA software application implementation for a particular subsystem task) assigned levels for each of the criteria and produces as output a single number or *utility* (u) suitable for comparing alternatives (e.g., for that subsystem task).

The main results of multiattribute utility theory cover *conditions* for which such utility functions can be expressed in simple mathematical forms and meaningfully and consistently calibrated using preference information. These tractable forms have parameters that are related to i) preferences for different levels of particular criteria (denoted by u_1, for criterion 1, u_i for the i'th criterion, etc.), and ii) tradeoffs between pairs of criteria. The key aspect of such preference models is that they are derived formally on a mathematically sound basis. The u_i and u are conventionally scaled to go between 0 (for the worst criteria levels) to 1 (for the best criteria levels).

These forms are flexible enough to reflect a variety of preferences adequately. The forms can be calibrated and combined to create an overall utility function for all the criteria for a specific task. In a similar vein, utilities for tasks can be aggregated to compute utilities for subsystems, etc. (Techniques for deriving function parameters from preference information are discussed in Reference 2.)

Illustrative Aggregating Across Criteria for a Task

For the Material Transactions subsystem, the following functional forms illustrate aggregating across criteria for a task:

$u_{task} = u_1 + u_{23456} - u_1 * u_{23456}$ (for an authorized individual, u_1 is set to 0)

$u_{23456} = u_{23} + u_{456} - u_{23} * u_{456}$

$u_{23} = u_2 + u_3 - u_2 * u_3$

$u_{456} = u_5 * (0.4 + 0.6 * u_{46})$

$u_{46} = 0.5 * u_4 + 0.5 * u_6$

If we examine the formula for u_{task} in this example, and imagine setting either u_1 or u_{23456} to 1.0 (corresponding to the best levels of the criteria involved), we see that u_{task} is equal to 1.0, the best possible score. The heuristic interpretation is a very simple one. An unauthorized insider must overcome *both* access and then subsequent "preventive barriers" and "mitigative controls" (auditing) to misuse the system. The best levels of either access control or the barrier/mitigative controls are sufficient to provide very high assurance that such misuse would not occur.

A similar heuristic applies to the formula u_{23456}. An authorized individual must overcome both barriers and mitigative controls to misuse the system, and if either barriers or mitigative controls are at their best criteria levels, they provide very high assurance against misuse. In this respect, the preference models above behave very much like probability models requiring an adversary to beat all safeguards components before compromising the system.

In this example, the formulas for the auditing criteria, however, do not have as direct a correspondence to probability models. The coefficients in the auditing criteria related formulas (u_{456}) essentially reflect preferences to account for the fact that even the lowest criterion levels of scrutability (attribute 4) or responsiveness (attribute 6) do not totally mitigative contributions from having auditing records with significant resolution (attribute 5).

With this illustrative preference function, we can evaluate different task design options such as the following three designs with the following criterion levels (and level utilities in parentheses) for the *modify* task:

	acc	aut	hum	scru	reso	resp
A:	3(.6)	2(.45)	1(0.0)	5(1.0)	5(1.0)	5(1.0)
B:	4(.75)	3(.75)	1(0.0)	4(.89)	5(1.0)	2(.47)
C:	2(.36)	2(.45)	5(1.0)	4(.89)	5(1.0)	1(0.0)

Using the formula for u_{task} the following utilities can be computed:

u_{task} (Design A) = 1.0

u_{task} (Design B) = 0.95 for authorized users, and 0.99 for unauthorized users.

u_{task} (Design C) = 1.0

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. All three designs provide relatively high assurance in spite of not having the best possible levels on many of the criteria. This is because being very strong in one area like preventive barriers can make up for being weaker in another area like mitigative controls and vice-versa. In the above example, facilities can have a variety of ways of achieving relatively high assurance without having to have every possible safeguards control implemented at a highest possible level. The aggregation illustrated here can provide insights to facilities as to the desirability of different combinations of safeguards controls for the insider threat.

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[3]. 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[4]. Figure 1 illustrates schematically how forms, applications, software, and procedures can be organized on a client-server system.

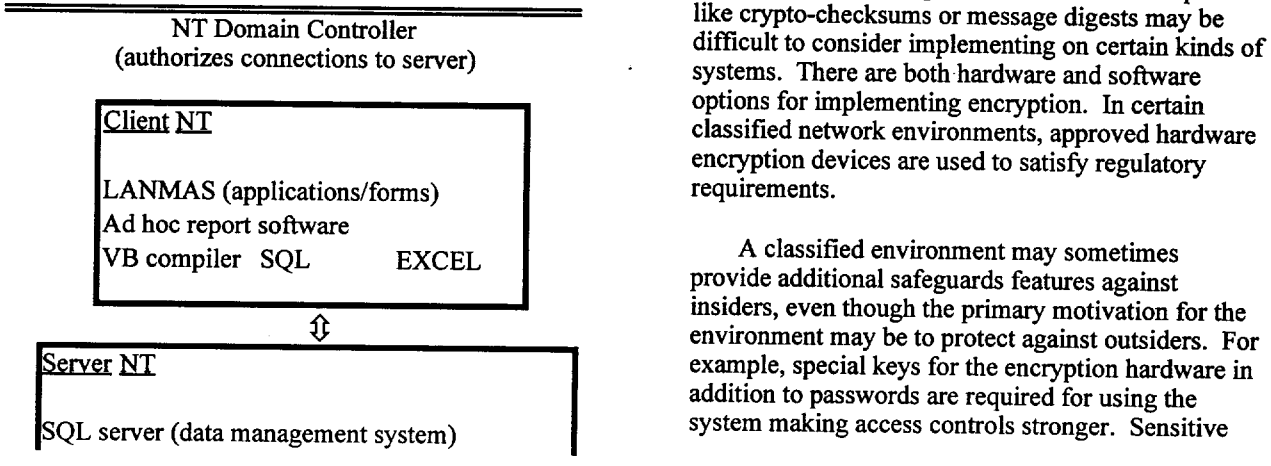


Figure 1. Schematic of Windows NT client-server organization

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.

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[5].

Data encryption in addition to password encryption is another safeguard 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. 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[4]. 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 Material Balance Areas or 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.

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[5].

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.

CURRENT PLANS AND SUMMARY

The methodology presented here is intended to be flexible to meet the needs of MA system planners and decision makers. The basic approach allows for easy modification or extension of details (such as the spectrum and gradations of criteria) to address decision making needs. Also, analysts can focus on selected tasks or one subsystem alone in an incremental analysis fashion. Current plans are to: continue testing and documenting the approach on LANMAS systems, demonstrate aggregated system evaluation, provide training and transfer of the approach to the field, more formally incorporate other objectives for comparing designs such as cost, operations and safety, develop a software aid for using the methodology (MATE - Material Accountability Threat Assessment), and extend the approach to other information based safeguards and security systems.

In summary, managers and policy makers must now cope with the rapidly changing MA environment of powerful personal computers on distributed networks, mixed security environments or moves to declassification, and computer-sophisticated insiders. We've developed a methodology that can help policy and decision makers:

- 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/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. 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).

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

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

4. 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

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

Assessing the Integrity of Local Area Network Material Accountability Systems Against Insider Threats

E. D. Jones and Alan Sicherman

Fission Energy and Systems Safety Program
Lawrence Livermore National Laboratory
Livermore, CA

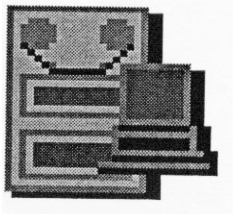
INMM 37th Annual Meeting
Naples, Florida

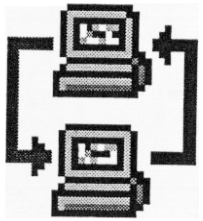
July 28 - 31, 1996

We're developing a methodology for analyzing insider threats against accountability applications

Applications include those residing on:

— Stand-alone mainframe platforms
— Client-server platforms
(e.g., LANMAS)





Our methodology can help managers and policy makers:

Collect information about applications relevant to the insider threat (info corruption --> hoax, cover-up, reduced assurance)

- Spotlight the relative strengths and weaknesses of application safeguards for various accountability functions

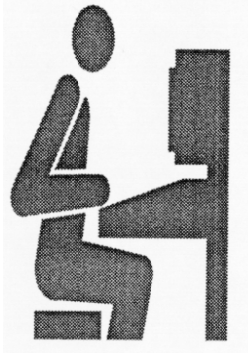
- Evaluate tradeoffs between different system/software designs for material accountability (MA) systems

Safeguards effectiveness depends on how diverse application functions/tasks are implemented

Client-server MA
Application System

Subsystems

- Inventories — Tasks
- Material Transactions — Tasks
- Report Generation — T
- Software Development — Tasks
- System Traffic — Tasks



Last year, we developed pragmatic criteria for analyzing MA application safeguards effectiveness

Safeguards criteria applied to each task:

Preventive Barriers	Mitigative
· Access/authorization	· Auditing
	- Scrutability
· Automated controls	- Resolution
· Human oversight	- Responsiveness

- Each criterion consists of a set of *explicitly described* safeguards levels or gradations for each MA subsystem
- The levels within each set are arranged from least preferable (weaker control) to most preferable (stronger control) from an insider threat perspective

Each criterion level has an explicit description tailored for each subsystem

Transactions subsystem simplified example:

<u>Audit - Responsiveness</u>

1 - only material anomaly <u>triggers</u> analysis of data available for audit

2 - anomaly and random checks

3 - anomaly and systematic audit of special sensitive transactions (e.g., *negative mass* transactions)

4 - anomaly & random checks & systematic audit of special sensitive actions

5 - all transactions proactively audited

The higher a criterion level, the more knowledge, skills, and abilities required by an insider to misrepresent information

Analyzing overall effectiveness requires aggregating across criteria, tasks and subsystems

Issues - how to quantify:

- 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

Key point to address - appropriate calibration of tradeoffs
- improving on one criterion versus another
- improving on one task versus another, etc.

Probabilities are used to quantify effectiveness in *traditional* vulnerability assessments or VA's. For accountability applications, other approaches (e.g., multiattribute utility/value) are more practical for calibrating tradeoffs.

A multiattribute utility (MAU) model quantifies the desirability of different safeguards combinations

An MAU model is developed in the following steps:

1) assess (quantify) preferences (from decision maker working with technical experts) for levels of each criterion (access, automated controls, human oversight, auditing criteria)

2) quantify relative importance of each criterion over its range by assessing tradeoffs of one criterion to gain on another

3) combine (in a consistent manner) the results of steps 1 and 2 to formulate a multiattribute utility function; the function computes a *utility* for any safeguards combination for a task; preferred combinations have higher utilities

Illustration - aggregating across criteria for a task with a multiattribute utility model

Material Transactions Subsystem

Modify Task (change accounting data, but no physical material actions)

	Access	Autom	Human	Scrutab	Audit Resol	Audit Resp	U_{Task}
Design A	3 (.6)	2 (.45)	1 (0)	5 (1.0)	5 (1.0)	5 (1.0)	1.0
Design B	4 (.75)	3 (.75)	1 (0)	4 (.89)	5 (1.0)	2 (.47)	.99 (.95)
Design C	2 (.36)	2 (.45)	5 (1.0)	4 (.89)	5 (1.0)	1 (0)	1.0

$U_{Task} = U_1 + U_{23456} - U_1 * U_{23456}$ (U_1= 0 for authorized)

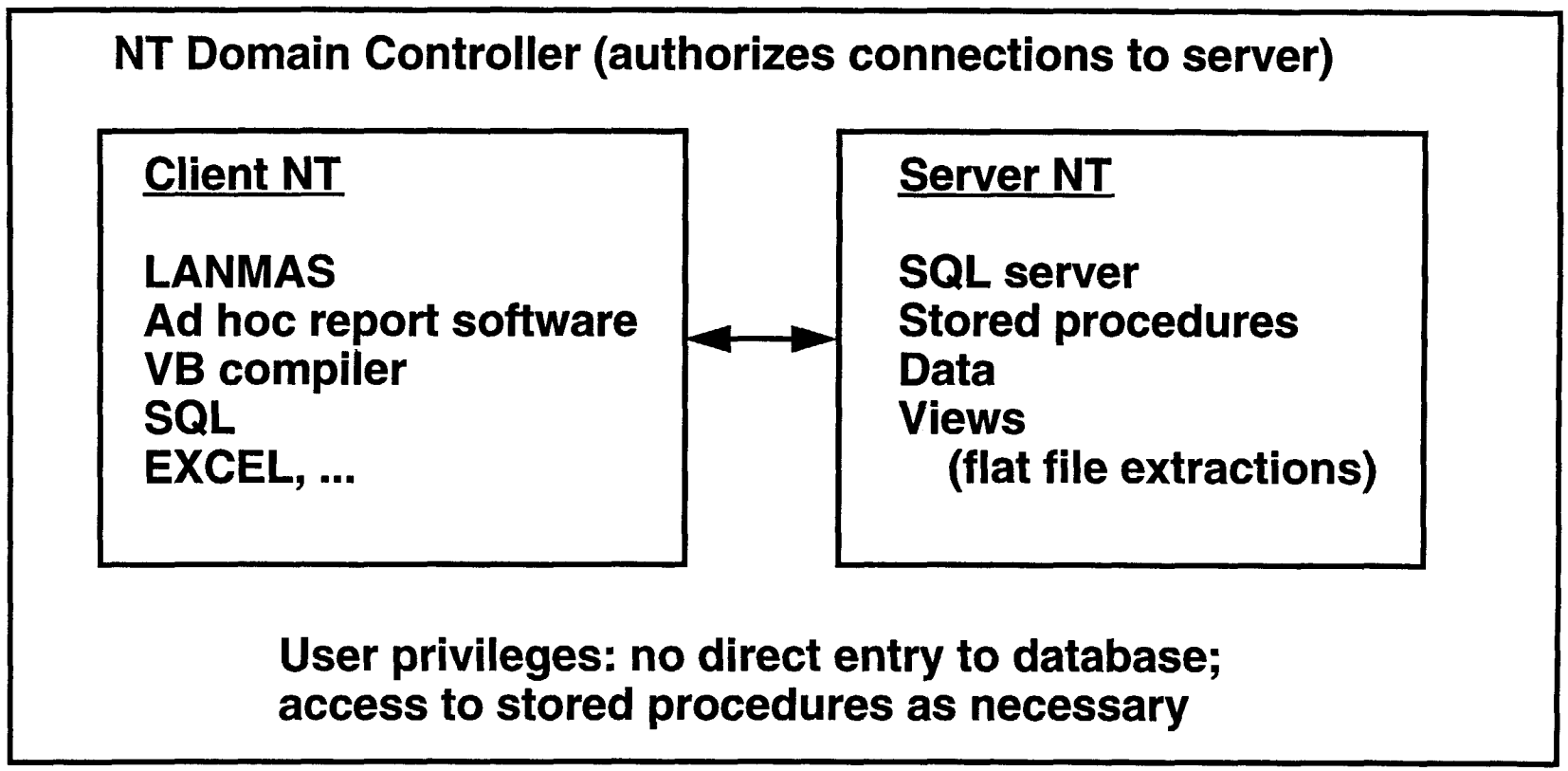
$U_{23456} = U_{23} + U_{456} - U_{23} * U_{456}$

$U_{23} = U_2 + U_3 - U_2 * U_3$

$U_{456} = U_5 * (0.4 + 0.6 * U_{46})$

$U_{46} = 0.5 * U_4 + 0.5 * U_6$

Methodology application to LANMAS prototypes on Windows NT



Methodology application to LANMAS prototypes

Dilbert strip

Methodology application to LANMAS prototypes

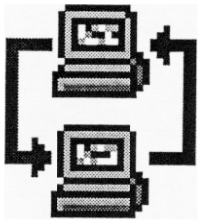
Considerations/insights which emerged:

- Access controls: Trojan Horses, encryption (sniffing) issues
- Automated controls: client vs server controls (flexibility vs security) {source code & core procedure mods issues; customized reports}
- Human oversight: no intrinsic provision for human oversight such as enforced two-person rule input; but *in transit* construct
- Scrutability: variety of querying and report routines possible to facilitate any auditing activity
- Resolution: transaction details preserved
- Responsiveness: great potential for proactive automated-assisted triggering of audits; BUT! need for it not necessarily recognized

Continuing and planned activities

- LANMAS evaluations
- Training and transfer of methodology
- Formally incorporate other objectives (cost, operations, safety)
- MATE (Material Accountability Threat Assessment) software aid
- Extension to other information based S&S systems

Our methodology can help decision makers cope with a rapidly changing MC&A environment

- Powerful PCs on distributed networks
- Mixed security environments or declassification
- Increasingly computer-sophisticated insiders
- Proliferation of types of computerized MA systems

We've developed a methodology to help evaluate tradeoffs among different system/software designs vis-à-vis effectiveness against the insider threat