NATIONAL COMPUTER

SECURITY CENTER

A GUIDE TO UNDERSTANDING CONFIGURATION MANAGEMENT

IN TRUSTED SYSTEMS

NCSC-TG-006-88 Library No. S-228,590

FOREWORD

This publication, "A Guide to Understanding Configuration Management in Trusted Systems", is being issued by the National Computer Security Center (NCSC) under the authority of and in accordance with Department of Defense (DoD) Directive 5215.1. The guidelines described in this document provide a set of good practices related to configuration management in Automated Data Processing (ADP) systems employed for processing classified and other sensitive information. Recommendations for revision to this guideline are encouraged and will be reviewed biannually by the National Computer Security Center through a formal review process. Address all proposals for revision through appropriate channels to:

National Computer Security Center 9800 Savage Road Fort George G. Meade, MD 20755-6000

Attention: Chief, Computer Security Technical Guidelines

_____ Patrick R. Gallagher, Jr. 28 March 1988 Director National Computer Security Center

ACKNOWLEDGEMENTS

Special recognition is extended to James N. Menendez, National Computer Security Center (NCSC), as project manager and primary author of this document.

Special acknowledgement is given to Grant Wagner, NCSC, and Dana Nell Stigdon, NCSC, for their constant help and guidance in the production of this document. Additionally, Dana Nell Stigdon, was responsible for writing the section on the Ratings Maintenance Program. Acknowledgement is also given to all those members of the computer security community who contributed their time and expertise by actively participating in the review of this document.

CONTENTS

(1) Unix is a registered trademark of Bell Laboratories

PREFACE

Throughout this guideline there will be recommendations made that are not included in the Trusted Computer System Evaluation Criteria (TCSEC) as requirements. Any recommendations that are not in the TCSEC will be prefaced by the word "should," whereas all requirements will be prefaced by the word "shall." It should be noted that a TCSEC rating will only be based upon meeting the TCSEC requirements. Recommendations are made in order to provide additional ways of increasing assurance. It is hoped that this will help to avoid any confusion.

1. PURPOSE

The Trusted Computer System Evaluation Criteria (TCSEC) is the standard used for evaluating the effectiveness of security controls built into ADP systems. The TCSEC is divided into four divisions: D, C, B, and A, ordered in a hierarchical manner with the highest division, A, being reserved for systems providing the best available level of assurance. Within divisions C through A are a number of subdivisions known as classes, which are also ordered in a hierarchical manner to represent different levels of security in these classes.

For TCSEC classes B2 through A1, the TCSEC requires that all changes to the Trusted Computing Base (TCB) be controlled by configuration management. Configuration management of a trusted system consists of identifying, controlling, accounting for, and auditing all changes made to the TCB during its development, maintenance, and design. The primary purpose of this guideline is to provide guidance to developers of trusted systems on what configuration management is and how it may be implemented in the development and life-cycle of a trusted system. This guideline has also been designed to provide guidance to developers of all systems on the importance of configuration management and how it may be implemented. Examples in this document are not to be construed as the only implementation that will satisfy the TCSEC requirement. The examples are merely suggestions of appropriate implementations. The recommendations in this document are also not to be construed as supplementary requirements to the TCSEC. The TCSEC is the only metric against which systems are to be evaluated.

This guideline is part of an on-going program to provide helpful guidance on TCSEC issues and the features they address.

2. SCOPE

An important security feature of TCSEC classes B2 through A1 is that there be configuration management procedures to manage changes to the Trusted Computing Base (TCB) and all of the documentation and tests affected by these changes. Additionally, it is recommended that such plans and procedures exist for systems not being considered for an evaluation or whose target evaluation class may be less than B2. The assurance provided by configuration management is beneficial to all systems. This guideline will discuss configuration management and its features as they apply to computer systems and products, with specific attention being given to those that are being built with the

intention of meeting the requirements of the TCSEC, and to those systems planning to be re-evaluated under the Ratings Maintenance Program (RAMP) (see Section 11. RAMP).

Except in cases where there is a distinction between the configuration management of a trusted system and an untrusted system, the word "system" shall be used as the object of configuration management, encompassing both the system and the TCB. It should be noted that the TCSEC only requires the TCB to be controlled by configuration management, although it is recommended that the entire system be maintained under configuration management.

3. CONTROL OBJECTIVES

The TCSEC gives the following as the Assurance Control Objective:

"Systems that are used to process or handle classified or other sensitive information must be designed to guarantee correct and accurate interpretation of the security policy and must not distort the intent of that policy. Assurance must be provided that correct implementation and operation of the policy exists throughout the system's life-cycle."[1]

Configuration management maintains control of a system throughout its life-cycle, ensuring that the system in operation is the correct system, implementing the correct security policy. The Assurance Control Objective as it relates to configuration management leads to the following control objective that may be applied to configuration management:

"Computer systems that process and store sensitive or classified information depend on the hardware and software to protect that information. It follows that the hardware and software themselves must be protected against unauthorized changes that could cause protection mechanisms to malfunction or be bypassed completely. [For this reason, changes to trusted computer systems, during their entire life-cycle, must be carefully considered and controlled to ensure that the integrity of the protection mechanism is maintained.] Only in this way can confidence be provided that the hardware and software interpretation of the security policy is maintained accurately and without distortion."[1]

4. ORGANIZATION

This document has been written to provide the reader with an understanding of what configuration management is and how it may be implemented in an ADP system.

For developers of trusted systems, this document also relates the TCSEC requirements to the configuration management practices that meet them. This document has been organized to illustrate the connection between practices and requirements through the use of a numbering convention for the TCSEC requirements. The configuration management requirements have been broken down into 19 separate requirements in Section 6 of this document. The requirement number(s) will be located in parenthesis following its appropriate discussion, e.g., (Requirements 2, 15), signifies that the previous discussion dealt with TCSEC requirements 2 and 15 as stated in Section 6.

5. OVERVIEW OF CONFIGURATION MANAGEMENT PRINCIPLES

Configuration management consists of four separate tasks: identification, control, status accounting, and auditing. For every change that is made to an automated data processing (ADP) system, the design and requirements of the changed version of the system should be identified. The control task of configuration management is performed by subjecting every change to documentation, hardware, and software/firmware to review and approval by an authorized authority. Configuration status accounting is responsible for recording and reporting on the configuration of the product throughout the change. Finally, through the process of a configuration audit, the completed change can be verified to be functionally correct, and for trusted systems, consistent with the security policy of the system. Configuration management is a sound engineering practice that provides assurance that the system in operation is the system that is supposed to be in use. The assurance control objective as it relates to configuration management of trusted systems is to "guarantee that the trusted portion of the system works only as intended."[1]

Procedures should be established and documented by a configuration management plan to ensure that configuration management is performed in a specified manner. Any deviation from the configuration

management plan could contribute to the failure of the configuration management of a system entirely, as well as the trust placed in a trusted system.

5.1 Purpose of Configuration Management

Configuration management exists because changes to an existing ADP system are inevitable. The purpose of configuration management is to ensure that these changes take place in an identifiable and controlled environment and that they do not adversely affect any properties of the system, or in the case of trusted systems, do not adversely affect the implementation of the security policy of the TCB. Configuration management provides assurance that additions, deletions, or changes made to the TCB do not compromise the trust of the originally evaluated system. It accomplishes this by providing procedures to ensure that the TCB and all documentation are updated properly.

6. MEETING THE CRITERIA REQUIREMENTS

This section lists the TCSEC requirements for configuration management. Each requirement for each class has been listed separately and numbered. Each number may be referenced to the requirement discussions that follow in this document. This section is designed to serve as a quick reference for TCSEC class requirements.

6.1 The B2 Configuration Management Requirements Requirement 1 - "During development and maintenance of the TCB, a configuration management system shall be in place."[1]

Requirement 2 - The configuration management system shall maintain "control of changes to the descriptive top-level specification (DTLS)."[1]

Requirement 3 - The configuration management system shall maintain control of changes to "other design data."[1]

Requirement 4 - The configuration management system shall maintain control of changes to "implementation documentation"[1] (e.g., user's manuals, operating procedures).

Requirement 5 - The configuration management system shall maintain control of changes to the "source code."[1]

Requirement 6 - The configuration management system shall maintain control of changes to "the running version of the object code."[1]

Requirement 7 - The configuration management system shall maintain control of changes to "test fixtures."[1]

Requirement 8 - The configuration management system shall maintain control of changes to test "documentation."[1]

Requirement 9 - "The configuration management system shall assure a consistent mapping among all documentation and code associated with the current version of the TCB."[1]

Requirement 10 - The configuration management system shall provide tools "for generation of a new version of the TCB from the source code."[1]

Requirement 11 - The configuration management system shall provide "tools for comparisons of a newly generated TCB version

with the previous version in order to ascertain that only the intended changes have been made in the code that will actually be used as the new version of the TCB."[1]

6.2 The B3 Configuration Management Requirements

The requirements for configuration management at TCSEC class B3 are the same as the requirements for TCSEC class B2. Although no additional requirements have been added, the configuration management system shall change to reflect changes in the design documentation requirements at class B3. This means that the additional documentation required for TCSEC class B3 shall also be maintained under configuration management.

6.3 The A1 Configuration Management Requirements

Requirements 2 through 11 are the same as those described in Section 6.1 for a class B2 rating. In addition the following requirements are added for class A1:

Requirement 12 - "During the entire life-cycle, i.e., during the design, development, and maintenance of the TCB, a configuration management system shall be in place for all security-relevant hardware, firmware, and software."[1]

Requirement 13 - The configuration management system shall maintain control of changes to the TCB hardware.

Requirement 14 - The configuration management system shall maintain control of changes to the TCB software.

Requirement 15 - The configuration management system shall maintain control of changes to the TCB firmware.

Requirement 16 - The configuration management system shall "maintain control of changes to the formal model."[1]

Requirement 17 - The configuration management system shall maintain control of changes to the "formal top-level specifications."[1]

Requirement 18 - The tools available for configuration management shall be "maintained under strict configuration control."[1]

Requirement 19 - "A combination of technical, physical, and procedural safeguards shall be used to protect from unauthorized modification or destruction the master copy or copies of all material used to generate the TCB."[1]

7. FUNCTIONS OF CONFIGURATION MANAGEMENT

7.1 Configuration Identification Configuration management procedures should enable a person to "identify the configuration of a system at discrete points in time for the purpose of systematically controlling changes to the configuration and maintaining the integrity and traceability of this configuration throughout the system life cycle."[4] The basic function of configuration identification is to identify the components of the design and implementation of a system. When it concerns trusted systems, this specifically means the design and implementation of the TCB. This task may be accomplished through the use of identifiers and baselines (see Section 9.1 The Baseline Concept). By establishing configuration items and baselines, the configuration of the system and its TCB can be accurately identified throughout the system life-cycle.

At TCSEC class B2, the TCSEC requires that "changes to the descriptive top-level specification, other design data, implementation documentation, source code, the running version of the object code, and test fixtures and documentation"[1] of the TCB be controlled by configuration management (Requirements 2, 3, 4, 5, 6, 7, 8). Configuration identification helps achieve this control. The TCSEC requires that each change to the TCB shall be individually identifiable so that a history of the TCB may be generated at any time. At TCSEC class A1, the requirements are extended to include that the "formal model...and formal top-level specifications" of the TCB shall also be maintained under the configuration management system (Requirements 16, 17).

The following is a sample list of what shall be identified and maintained under configuration management:

* the baseline TCB including hardware, software, and firmware

* any changes to the TCB hardware, software, and firmware since the previous baseline

* design and user documentation

* software tests including functional and system integrity tests

* tools used for generating current configuration items (required at TCSEC class A1 only)

Configuration management procedures should make it possible to accurately reproduce any past TCB configuration. In the event a

security vulnerability is discovered in a version of the TCB other than the most current one, analysts will need to be able to reconstruct the past environment. This reconstruction will be possible to perform if proper configuration identification has been performed throughout the system life-cycle.

The TCSEC also requires at class B2 and above, that tools shall be provided "for generation of a new version of the TCB from the source code" and that there "shall be tools for comparing a newly generated version with the previous TCB version in order to ascertain that only the intended changes have been made in the code that will actually be used as the new version of the TCB"[1] (Requirements 10, 11).

These tools are responsible for providing assurance that no additional changes have been inserted into the TCB that were not intended by the system designer. Automated tools are available that make it possible to identify changes to a system online (see APPENDIX A: AUTOMATED TOOLS). Any changes, or suggested changes to a system should be entered into an online library. This data can later be used to compare any two versions of a system. Such online configuration libraries may even provide the capability for line-by-line comparison of software modules and documentation. At Class A1, the tools used to perform this function shall be "maintained under strict configuration control"[1] (Requirement 18). These tools shall not be changed without having to undergo a strict review process by an authorized authority.

7.1.1 Configuration Items

A configuration item is an uniquely identifiable subset of the system configuration that represents the smallest portion of the system to be subject to independent configuration management change control procedures. Configuration items need to be individually controlled because any change to a configuration item may have some effect upon the properties of the system or the security policy of the TCB.

Configuration items as they relate to the TCB, are subsets of the TCB's hardware, firmware, software, documentation, tests, and at class A1, development tools. Each module of TCB software for example, may constitute a separate configuration item. Configuration items should be assigned unique identifiers (e.g., serial numbers, names) to make them easier to identify throughout the system life-cycle. Proper identification plays a vital role in meeting the TCSEC requirement for class B2 that requires the configuration management system to "assure a consistent mapping among all documentation and code associated with the current version of the TCB"[1] (Requirement 9). Used in conjunction with

a configuration audit, a consistent labeling system helps tie documentation to the code it describes. Not only does labeling each configuration item make them easier to identify, but it also increases the level of control that may be maintained over the entire system by making these items more traceable.

Configuration items may be given an identifier through a random distribution process, but, it is more useful for the configuration identifier to describe the item it identifies. Selecting different fields of the configuration identifier to represent characteristics of the configuration item is one method of accomplishing this. The United States Social Security number is a "configuration identifier" we all have that uses such a system. The different fields of the number identify where we applied for the Social Security card, hence describing a little bit about ourselves. As the configuration identifier relates to computer systems, one field should identify the system version the item belongs to, the version of software that it is, or its interface with other configuration items. When using a numbering scheme like this, a change to a configuration item should result in the production of a new configuration identifier. This new identifier should be produced by an alteration or addition to the existing configuration identifier. A new version of a software program should not be identified by the same configuration item number as the original program. By treating the two versions as distinct configuration items, line- by-line comparisons are possible to perform.

Identifying configuration items is a task that should be performed early in the development of the system, and once something is designated as a configuration item, the design of that item should not change without the knowledge and permission of the party controlling the item. Early identification of configuration items increases the level of control that may be maintained over the item and allows the item to be traced back through all stages of the system development. In the event that a configuration item is not identified until late in the development process, accountability for that item in the early stages of the system development would be non-existent.

Configuration items may vary widely in complexity, size, and type, and it is important to choose configuration items with appropriate granularity. If the items are too large, the data identifying each one will overwhelm anyone trying to audit the system. If the items are too small, the amount of total identification data will overwhelm the system auditors.[2] The appropriate granularity for configuration items should be identified by each vendor and documented in the configuration management plan.

7.2 Configuration Control

"Configuration control involves the systematic evaluation, coordination, approval, or disapproval of proposed changes to the design and construction of a configuration item whose configuration has been formally approved."[5] Configuration control should begin in the earliest stages of the design and development of the system and extend over the full life of the configuration items included in the design and development stages. Early initiation of configuration control procedures provides increased accountability for the system by making its development more traceable. The traceability function of configuration control serves a dual purpose. It makes it possible to evaluate the impact of a change to the system and controls the change as it is being made. With configuration control in place, there is less chance of making undesirable changes to a system that may later adversely affect the security of the system.

Initial phases of configuration control are directed towards control of the system configuration as defined primarily in design documents. For these, the Configuration Management plan shall specify procedures to ensure that all documentation is updated properly and presents an accurate description of the system and TCB configuration. Often a change to one area of a system may necessitate a change to another area. It is not acceptable to only write documentation for new code or newly modified code, but rather documentation for all parts of the TCB that were affected by the addition or change shall be updated accordingly. Although documentation may be available, unless it is kept under configuration management and updated properly it will be of little, if any use. In the event that the system is found to be deficient in documentation, efforts should be made to create new documentation for areas of the system where it is presently inadequate or non-existent.

To meet the TCSEC requirements though, configuration control shall cover a broader area than just documentation, and at Class B2 shall also maintain control of "design data, source code, the running version of the object code, and test fixtures"[1] of the TCB (Requirements 3, 5, 6, 7). A change to any of these shall be subject to review and approval by an authorized authority.

For TCB configuration items, those items shall not be able to change without the permission of the controlling party. At TCSEC class A1, this requirement is strengthened to require "procedural safeguards"[1] to protect against unauthorized modification of the materials used in the TCB (Requirement 19). These procedures should require that not only does the

controlling party need to give permission to have a change performed, but that the controlling party performs the change on the master copy of the TCB that will be released. This ensures against changes being made to the master copy that are different than the approved changes.

The degree of configuration control that is exercised over the TCB will affect whether or not it meets the TCSEC requirements for configuration management. The configuration management requirements in the TCSEC require that a configuration management system be in place during the "development and maintenance of the TCB" at Class B2 (Requirement 1), and at Class A1, "during the entire life-cycle"[1]

of the TCB (Requirement 12). A minimal configuration control system that would not be sufficient in meeting the TCSEC requirements, may only provide for review after a change has been made to the system. A system such as this may ensure that the change is complete and acceptable and may control the release of the change, but for the most part, the control exercised is little more than an after-the-fact quality assurance check. This system is certainly better than having no control system in place, but it would not meet the TCSEC requirements for configuration management. What is missing from this system that would bring it closer to the B2 requirements is control over the change as it is being made. The configuration control required by the TCSEC should provide for constant checking and approval of a change from its inception, through implementation and testing, to release. The level of control exercised over the TCB may exceed that of the rest of the system, but it is recommended that all parts of the system be under configuration control.

In the case of a change to hardware or software/firmware that will be used at multiple sites, configuration control is also responsible for ensuring that each site receives the appropriate version of the system.

The point behind configuration control of the TCB is that all changes to the TCB shall be approved, monitored, and evaluated to provide assurance that the TCB functions properly and that all security policies are maintained.

7.3 Configuration Status Accounting

Configuration status accounting is charged with reporting on the progress of the development in very specific ways. It accomplishes this task through the processes of data recording, data storing, and data reporting. The main objective of configuration status accounting is to record and report all information that is of significance to the configuration

management process. What is of significance should be outlined in the Configuration Management Plan. The establishment of a new baseline (see Section 9.1 THE BASELINE CONCEPT) or the meeting of a milestone is an example of what should be recorded as configuration status accounting information. The requirements in the configuration management plan should be viewed as the minimum and any events that seem relevant to configuration management should be captured and recorded in that they may prove to be useful in the future.

The configuration accounting system may consist of tracing through documentation manually to find the status of a change or it may consist of a database that can automatically track a change. As long as the information exists accurately in some form though, it will serve its purpose. The benefit of an online status accounting system is that the information may be kept in a more structured fashion, which would facilitate keeping it up to date. Being able to query a database for information concerning the status of a configuration change or configuration item would also be less cumbersome than sorting through notebook pages. Finally, the durability of a diskette or hard disk for storage outweighs that of a spiral notebook or folder, provided that it is properly backed up to avoid data loss in the event of a system failure.

Whichever system is used, it should be possible to quickly locate all authorized versions of a configuration item, add together all authorized changes with comments about the reason for the change, and arrive at either the current status of that configuration item, or some intermediate status of the requested item. The status of all authorized changes being performed should be formulated into a System Status Report that will be presented at a Configuration Control Board meeting (see Section 9.3 THE CONFIGURATION CONTROL BOARD).

Configuration status accounting "establishes records and reports which enable proper logistics support, i.e., the supplying of spares, instruction manuals, training and maintenance facilities, etc. to be established."[5] The records and reports produced through configuration status accounting should include a current configuration list, an historical change list, the original designs, the status of change requests and their implementation, and should provide the ability to trace all changes.

7.4 Configuration Audit

Configuration auditing involves checking for top to bottom completeness of the configuration accounting information "to

ascertain that only the [authorized] changes have been made in the code that will actually be used as the new version of the TCB."[1] (Requirement 11) When a change has been made to a system, it should be reviewed and audited for its effect on the rest of the system. This should include reviewing and testing all software to ensure that the change has been performed correctly.

Configuration auditing is concerned with examining the control process of the system and ensuring that it actually occurs the way it should. Configuration auditing for trusted systems verifies that after a change has been made to the TCB, the security features and assurances are maintained. Configuration audits should be performed periodically to verify the configuration status accounting information. The configuration audit minimizes the likelihood that unapproved changes have been inserted without going unnoticed and that the status accounting information adequately demonstrates that the configuration management assurance is valid.

"A complete audit should include tracing each requirement down through all functions that implement it to see if that requirement is met."[2] Furthermore, the configuration audit should also ensure that no additions were made that were not required. For the audit to provide a useful form of technical review, it should be predictable and as foolproof as possible, i.e., there should be specific desired results.

The configuration audit should verify that:

* the architectural design satisfies the requirements

* the detailed design satisfies the architectural design

* the code implements the detailed design

* the item/product performs per the requirements

* the configuration documentation and the item/product match

The main emphasis of configuration auditing is on providing the user with reasonable assurance that the version of a system in use is the same version that the user expects to be in use. Configuration audits ensure that the configuration control procedures of the configuration management system are being followed. The assurance feature of configuration auditing is provided through reasonable and consistent accountability procedures. All code audits should follow roughly the same procedures and perform the same set of checks for every change to the system.

8. THE CONFIGURATION MANAGEMENT PLAN

Effective configuration management should include a well-thought- out plan that should be prepared immediately after project initiation. This plan should describe, in simple, positive statements, what is to be done to implement configuration management in the system and TCB. A minimal configuration management plan may be limited to simply defining how configuration management will be implemented as it relates to the identification, control, accounting, and auditing tasks. The configuration management plan described in the following paragraphs is an example of a plan that goes into more detail and contains documentation on all aspects of configuration management, such as examples of documents to be used for configuration management, procedures for any automated tools available, or a Configuration Control Board roster (see Section 9.3 THE CONFIGURATION CONTROL BOARD). The configuration management plan should contain documentation that describes how the configuration management "tasks are to be carried out in sufficient detail that anyone involved with the project can consult them to determine how each specific development task relates to CM."[2]

One portion of the configuration management plan should define the roles played by designers, developers, management, the Configuration Control Board, and all of the personnel involved with any part of the life-cycle of the system. The responsibilities required by all those involved with the system should be established and documented in the configuration management plan to ensure that the human element functions properly during configuration management. A list of Configuration Control Board members, or the titles of the members should also be included in this section.

Any tools that will be available and used for configuration management should be documented in the configuration management plan. At TCSEC class A1, it is required that these tools shall be "maintained under strict configuration control"[1] (Requirement 18). These tools may include forms used for change control, conventions for labeling configuration items, software libraries, as well as any automated tools that may be available to support the configuration management process. Samples of any documents to be used for reporting should also be contained in the configuration management plan with a description of each.

A section of the Configuration Management Plan should deal with procedures. Since the main thrust of configuration management consists of the following of procedures, there needs to be thorough documentation on what procedures one should follow

during configuration management. The configuration management plan should provide the procedures to take to ensure that both user and design documentation are updated in synchrony with all changes to the system. It should include the guidelines for creating and maintaining functional tests and documentation throughout the life of the system. The configuration management plan should describe the procedures for how the design and implementation of changes are proposed, evaluated, coordinated, and approved or disapproved. The configuration management plan should also include the steps to take to ensure that only those approved changes are actually included and that the changes are included in all of the necessary areas.

Another portion of the configuration management plan should define any existing "emergency" procedures, e.g., procedures for performing a time sensitive change without going through a full review process, that may override the standard procedure. These procedures should define the steps for retroactively implementing configuration management after the emergency change has been completed.

The configuration management plan is a living document and should remain flexible during design and development phases. Although the configuration management plan is in place to impose control on a project, it should still be open to additions and changes as designers and developers see fit. This is not to say that the configuration management plan is only a guide and need not be followed, but that modifications should be able to occur. If the plan is not followed, there is no way it will be able to provide the appropriate assurances. In the event that a change is needed to the configuration management plan, the change should be carefully evaluated and approved. In changes to the configuration management plan of a trusted system this evaluation shall ensure that the security features and assurances supported by the plan are still maintained after the change has been implemented.

9. IMPLEMENTATION METHODS

This section discusses implementation methods for configuration management that may be used to meet some of the requirements of the TCSEC. Section 9.1 discusses the baseline concept as a method of configuration identification. The baseline concept utilizes the features of configuration management spoken of previously, but divides the life-cycle of the system into different baselines.

Section 9.2 illustrates how a fictitious company, MER, Inc., conducts configuration management. They are attempting to meet the TCSEC requirements for a B2 system.

Section 9.3 discusses the concept of a Configuration Control Board (CCB) for carrying out configuration control. A CCB is a body of people responsible for configuration control. This concept is widely used by many computer vendors.

9.1 The Baseline Concept

Baselines are established at pre-selected design points in the system life-cycle. One baseline may be used to describe a specific version of a system, or in some configuration management systems a single baseline may be defined at each of several major milestones. Baselines should be established at the discretion of the Configuration Control Board and outlined in the configuration management plan. In cases where several baselines are established, each baseline serves as a cutoff point for one segment of development, while simultaneously acting as the step off point for another segment. The characteristics common to all baselines are that the design of the system will be approved at the point of their establishment and it is believed that any changes to this design will have some impact on the future development of the system.

Baseline management is one technique for performing configuration identification. It identifies the system and TCB design and development as a series of phases or baselines that are subject to configuration

control. Used in conjunction with configuration items, this is another effective way to identify the system and its TCB configuration throughout its life-cycle.

"For each different type of baseline, the individual components to be controlled should be identified, and any changes that update the current configuration should be approved and documented. For each intermediate product in the development [life-cycle] there is only one baseline. The current

configuration can be found by applying all approved changes to the baseline."[2]

In a system defining several baselines for different stages of development, these baselines or milestones should be established at the system inception to serve as guides throughout the development process. Although specific baselines are established in this case, alternatives may be recommended to promote greater design flexibility or efficiency. The number of baselines that may be established for a system will vary depending upon the size and complexity of the system and the methods supported by the designers and developers. It is possible to establish multiple baselines existing at the same time so long as configuration management practices are applied properly to each baseline. The following example will discuss the baseline concept using three common baseline categories: functional, allocated, and product. It should be emphasized that these are simply basic milestones and baselines should be established depending upon the decisions of the designers and developers.

The first baseline, the functional baseline, is established at the system inception. It is derived from the performance and objectives criteria documentation that consists of specifications defining the system requirements. Once these specifications have been established, any changes to them should be approved.

The requirements produced in the functional baseline may be divided and subdivided into various configuration items. Once it has been decided what the configuration items will be, each of the items should be given a configuration identifier. From the analysis of the system requirements the allocated baseline will be established. This baseline identifies all of the required functions with a specific configuration item that is responsible for the function. In this baseline, an individual should be charged with the responsibility for each configuration item. All changes affecting specifications defining design requirements for the system or its configuration items as stated in the allocated baseline should require approval of the responsible individual.

The final baseline, the product baseline, should contain that version of the system that will be turned over for integration testing. This baseline signifies the end of the development phase and should contain a releasable version of the system.

The baseline example mention earlier in which one baseline is established for a single version of a system entails the same reasoning as the functional, allocated, and product baseline

example. The system established as a baseline in the single baseline example will need to have an approved design before being placed under configuration control. Prior to the design approval, the system design will have to have undergone some type of functional review and a process that would allocate these functions to various configuration items. Although the early processes of the design will not be as formal in the single baseline example as they are when the early tasks are individually defined, the system will still benefit from being under the control of configuration management as a baseline. The main point of

establishing any baseline is controlling changes to that baseline by requiring any changes to it to have to undergo an established change control process.

9.2 Configuration Management at MER, Inc.

MER, Inc., is a manufacturer of computer systems. Their latest project consists of building a system that will meet the B2 requirements of the TCSEC. In the past, their configuration management has only consisted of quality assurance checks, but to meet the B2 requirements they realize that they will need to have specific configuration management procedures in place during the development and maintenance of the system.

The project manager was assigned the task of writing the configuration management procedures and elected to present them in a configuration management plan. After doing some research on what should be contained in the configuration management plan, he proceeded to write a plan for MER, Inc. The configuration management plan that was written listed all of the steps to be followed when carrying out configuration management for the system. It described the procedures to be followed by the development team and described the automated tools that were going to be used at MER, Inc. for configuration management. These tools consisted of an online tracking data base to be used for status accounting, an online data base that contained a listing of all of the items under configuration control, and automated libraries used for storing software. Before development began, all of the development team was responsible for reading the configuration management plan to ensure that they were aware of the procedures to be followed for configuration management.

As the system was developed, the TCB hardware, software, and firmware were labeled using a configuration item numbering scheme that had been explained in the configuration management plan. In addition, the documentation and tests accompanying these items were also given configuration item numbers to assure a consistent

mapping between TCB code and these items. All of the configuration item numbers and a description of the items were stored in a data base that could be queried at any time to derive the configuration of the entire system. Software and documentation were stored in a software library where they could be retrieved and worked on without affecting the master versions. The master copies of all software were stored in a master library that contained the releasable versions of the software. Both of these libraries are protected by a discretionary access control mechanism to prevent any unauthorized personnel from tampering with the software.

During the development of the system, changes were required. The procedures for performing a change under configuration control are described in the configuration management plan. These are the same procedures that will remain in effect throughout the life-cycle of the system. For each proposed change, a decision has to be made by management whether or not the change is feasible and necessary. MER, Inc. has an online forum for reviewing suggested changes. This forum makes it possible for all of the members of the development team to comment on how the proposed change may affect their work. Management would often consult this forum to help arrive at their final decision.

After a decision was made, a programmer was assigned to perform the change. The programmer would retrieve the most recent version of the software from the software library and proceed to change it. As the change was being performed, the changes were entered into the online tracking data base. This made it possible for members of the development team to query this data base to find the current status of the change at any time. After the change had been performed it was tested and documented, and upon successful completion it was forwarded to a reviewer. This reviewer was the software manager, who was

the only person authorized to approve a changed version for release. After the change was approved for release, the changed version was stored in the master library and a second copy was stored in the software library. Each change stored in these libraries was given a new configuration identification number. A tool was available at MER, Inc. that made it possible to identify changes made to software. It compared any two versions of the software and provided a line-by-line listing of the differences between the two.

It was realized at the beginning of the development process that there would be times when critical changes would need to be performed that would not be able to undergo this review process. For these changes, emergency procedures had been listed in the configuration management plan and a critical fix library was

available to record critical changes that had occurred since a release.

A control process for changes to the TCB hardware was also provided for in the configuration management plan. The procedures ensured that changes to the TCB hardware were traceable and did not violate the security assumptions made by the TCB software. Similar to software changes, all hardware changes were reviewed by the project manager before being implemented.

After a change is made to the TCB software, MER, Inc. performs a configuration audit to verify the information that exists in the tracking data base. Whether or not a change is performed, the configuration management plan at MER, Inc. specifies that a configuration audit be performed at least once a month. This audit compares the current master version with the status accounting information to verify that no changes have been inserted that were not approved.

This configuration management plan encompasses the descriptive top-level specification (DTLS), implementation documentation, source code, object code, test fixtures, and test documentation, and has been found to satisfy the TCSEC requirements for configuration management at class B2.

9.3 The Configuration Control Board (CCB)

Configuration control may be performed in different ways. One method of configuration control that is in use by systems already evaluated at TCSEC Class B2 and above is to have the control carried out by a body of qualified individuals known as the Configuration Control Board (CCB), also known as the Configuration Change Board. The Board is headed by a chairperson, who is responsible for scheduling meetings and for giving the final approval on any proposed changes. The membership of the CCB may vary in size and composition from organization to organization, but it should include members from any or all of the following areas of the system team:

* Program Management

* System Engineering

* Quality Assurance

* Technical Support

* Integration and Test

* System Installation

* Technical Documentation

* Hardware and Software/Firmware Acquisition

* Program Development

* Security Engineering

* User Groups

The members of the CCB should interact periodically, either through formal meetings, electronic forums, or any other available means, to discuss configuration management topics such as proposed changes, configuration status accounting reports, and other topics that may be of interest to the different areas of the system development. These interactions should be held at periodic intervals to keep the entire system team up-to-date with all advancements or alterations in the system. The Board serves to control changes to the system and ensures that only approved changes are implemented into the system. The CCB carries out this function by considering all proposals for modifications and new acquisitions and by making decisions regarding them.

An important part of having cross representation in the CCB from various groups involved in the system development is to prevent "unnecessary and contradictory changes to the system while allowing changes that are responsive to new requirements, changed functional allocations, and failed tests."[2] All of the members of the Board should have a chance to voice their opinions on proposed changes. For example, if system engineering proposes a change that will affect security, both sides should be able to present their case at a CCB meeting. If diversity did not exist in the CCB, changes may be performed, and upon implementation may be found to be incompatible with the rest of the system.

The configuration control process begins with the documentation of a change request. This change request should include justification for the proposed change, all of the affected items and documents, and the proposed solution. The change request should be recorded, either manually or online in order to provide a way of tracking all proposed changes to the system and to ensure against duplicate change requests being processed.

When the change request is recorded, it should be distributed for analysis by the CCB who will review and approve or disapprove the

change request. An analysis of the total impact of the change will decide whether or not the change should be performed. The CCB will approve or disapprove the change request depending upon whether or not the change is viewed as a necessary and feasible change that will further the design goals of the system. In situations where trusted systems are involved, the CCB shall also ensure that the change will not affect the security policy of the system.

Once a decision has been reached regarding any modifications, the CCB is responsible for prioritizing the approved modifications to ensure that those that are most important are developed first. When prioritizing changes, an effort should be made to have the changes performed in the most logical order whenever possible. The CCB is also responsible for assigning an authority to perform the change and for ensuring

that the configuration documentation is updated properly. The person assigned to do the change should have the proper authorization to modify the system, and in trusted systems processing sensitive information, this authorization shall be required. During the development of any enhancements and new developments, the CCB continues to exert control over the system by determining the level of testing required for all developments.

Upon completion of the change, the CCB is responsible for verifying that the change has been properly incorporated and that only the approved change has been incorporated. Tests should be performed on the modified system or TCB to ensure that they function properly after the change is completed. The CCB should review the test results of any developments and should be the final voice on release decisions.

The use of a CCB is one way of performing configuration control, but not every vendor may have the desire or resources to establish one. Whatever the preference, there should still be some way of performing the control processes described previously.

10. OTHER TOPICS

10.1 Trusted Distribution

Related to the configuration management requirements for trusted systems is the TCSEC requirement for trusted distribution at class A1 which states:

"A trusted ADP system control and distribution facility shall be provided for maintaining the integrity of the mapping between the master data describing the current version of the TCB and the on-site master copy of the code for the current version. Procedures (e.g., site security acceptance testing) shall exist for assuring that the TCB software, firmware, and hardware updates distributed to a customer are exactly as specified by the master copies."[1]

Two questions that the trusted distribution process should answer are: (a) Did the product received come from the organization who was supposed to have sent it? and (b) Did the recipient receive exactly what the sender intended?

Configuration management assists trusted distribution by ensuring that no alterations are made to the TCB from the time of approved modification to the time of release. The additional configuration management requirement at A1 that supports this is, "A combination of technical, physical and procedural safeguards shall be used to protect from unauthorized modification or destruction the master copy or copies of all material used to generate the TCB"[1] (Requirement 19). This requirement calls for strict control over changes made to any versions of the TCB. The possibility that a change may not be

performed as specified, or that a harmful modification may be inserted into the TCB should be considered and the authority to perform changes to the master copy should be restricted. A single master copy authority should be made responsible for ensuring that only approved and acceptable changes are implemented into the master copy.

Configuration status accounting records and auditing reports can provide accountability for all TCB versions in use. In the event of altered copies being distributed or "bogus" copies being distributed that were not manufactured by the vendor, configuration management records will be able to assess the validity and accuracy of all TCB versions. Trusted distribution displays the need for configuration control over all changes to the TCB. Without configuration control there would be no accountability for the TCB versions distributed to the customer.

10.2 Functional Testing

"The system developer shall provide to the evaluators a document that describes the test plan, test procedures that show how the security mechanisms were tested, and results of the security mechanisms' functional testing."[1] The creation and maintenance of these functional tests is required to be part of the configuration management procedures. Test results and any affected test documentation shall be maintained under configuration management and updated wherever necessary (Requirements 7, 8). The tests should be repeatable, and include sufficient documentation so that any knowledgeable programmer will be able to figure out how to run them. The test plan for the system should be described in the functional specification (or other design documentation) for the TCB, along with descriptions of the test programs. The test plan and programs should be reviewed and audited along with the programs they test, although the coding standards need not be as strict as those of the tested programs.

It is not acceptable to only generate tests for code that was opened or replaced, but all of the portions of the TCB that were affected by the change should also be tested. The NCSC evaluators can provide a description of the security functional tests required to meet the TCSEC testing requirements, including the testing required as stated above for configuration management.

10.3 Configuration Management Training

Each new technical employee should receive training in the configuration management procedures that a particular installation follows. Experienced programmers, although they may be familiar with some form of configuration management, will also require training in any new procedures, i.e., an automated accounting system, that will be required to be followed. Training should be conducted either "by holding formal classes or by setting aside sufficient time for the reading of the company wide configuration standards."[2] New programmers should become familiar with the Configuration Management Plan before being allowed to incorporate any changes into the design baseline. It should be stressed that a failure to maintain the configuration management standards resulting from untrained employees, could prevent the system from receiving a rating.[2]

10.4 Configuration Management Supervision A successful configuration management system requires the following of many procedures. Considering the demands made on the system staff, errors may occur and shortcuts may be sought which will jeopardize the entire configuration management plan. A review process should be present to ensure that no single person can create a change to the system and implement it without being subject to some type of approval process. Supervisors, who are responsible for the personnel performing the change should be required to sign an official record that the change is the correct change.[2]

Proper supervision also provides assurance that whoever performs the change has the proper authorization to do so. Changes should not be performed by personnel that are not qualified to perform the change. Also, in systems that process sensitive information, the programmer performing the change shall possess the proper security clearance to perform the change.

Management itself must directly support the configuration management plan in order for it to work. It should not encourage cutting configuration management corners under any circumstances, e.g., due to scheduling or budgeting. Management should be willing to support the expenditure of money, people, and time to allow for proper configuration management.

11. RATINGS MAINTENANCE PROGRAM

The Ratings Maintenance Program (RAMP) has been developed by the NCSC in an effort to keep the Evaluated Products List (EPL) current. By training vendor personnel to recognize which changes may adversely affect the implemetation of the security policy of the system, and to track these changes to the evaluated product through the use of configuration management, RAMP will permit a vendor to maintain the rating of the evaluated product without having to re-evaluate the new version. Because changes from one version of an operating system to the next version may affect the security features and assurances of that operating system, configuration management is an integral part of RAMP. For a system to maintain its rating under this program, the NCSC shall be assured, through the vendor's configuration management

procedures, that the changes made have not adversely affected the implementation of the security mechanisms and assurances of the system.

Each RAMP participant shall develop an NCSC approved Rating Maintenance Plan (RMPlan) which includes a detailed Configuration Management Plan (CMP) to support the rating maintenance process. This requirement applies to all systems participating in RAMP, regardless of class. For further information about the RAMP program and about configuration management requirements for RAMP, contact:

National Computer Security Center 9800 Savage Road Fort George G. Meade, MD 20755©6000
Attention: Chief, Requirements and Resources Division

12. CONFIGURATION MANAGEMENT SUMMARY

The assurance provided by configuration management is beneficial to all systems. It is a requirement for trusted systems for classes B2 and above that a configuration management system "be in place that maintains control of changes to the descriptive top-level specification, other design data, implementation documentation, source code, the running version of the object code, and test fixtures and documentation"[1] (Requirements 1, 2, 3, 4, 5, 6, 7, 8). Although configuration management is a requirement for trusted systems for classes B2 and above, it should be in place in all systems regardless of class rating, or if the system has a rating at all.

Successful configuration management is built around four main objectives: control, identification, accounting, and auditing. Through the accomplishment of these objectives, configuration management is able to maintain control over the TCB and protect it against "unauthorized changes that could cause protection mechanisms to malfunction or be bypassed completely."[1] Even for those aspects of the system which are not security-relevant, configuration management is still a valuable method of ensuring that all of the properties of a system are maintained after a change. It is very important to the success of configuration management that a formal configuration management plan be adhered to during the life-cycle of the system.

A successful configuration management plan should begin with early and complete definition of configuration management goals, scope, and procedures. The success of configuration management is

dependent upon accuracy. Changes should be identified and accounted for accurately, and after the change is completed, the change, and all affected parts of the system should be thoroughly documented and tested.

Configuration management provides control and traceability for all changes made to the system. Changes in progress are able to be monitored through configuration status accounting information in order to control the change and to evaluate its impact on other parts of the system.

An important part of having a successful configuration management plan is that the people involved with it must adhere to its procedures in order to keep all documentation current and the status of changes up-to-date.

With a firm and well documented configuration management plan in place, the occurrence of any unnecessary or duplicate changes will be reduced greatly and any necessary changes that are

required should be able to be identified with great ease. An effective configuration management system should be able to show what was supposed to have been built, what was built, and what is presently being built.

APPENDIX A: AUTOMATED TOOLS

Automated tools may be used to perform some of the configuration management functions that previously had to be performed manually. A data base management system, even with just a limited query system, may be used to perform the configuration audit and status accounting functions of configuration management. The principle behind using automated systems is that text, both from source code and other documents involved in the development of the system, can be entered into a Master Library and modified only through the use of the automated system. This prevents anyone from performing a change without having the proper authorization to access the configuration data base. "In general, only one program librarian, who should be the project manager or someone directly responsible to the manager, should have write access to the Master Library during development."[2]

A number of software developers have created software control facilities that are currently available to be used for configuration status accounting. A brief discussion of two of these systems follows.

A.1 UNIX (1) SCCS

"Under the Unix (1) system, the make utility, and the elements admin, get, prs, and delta, which comprise the Source Code Control System, provide a basic configuration accounting system. Initially a directory is created using the mkdir function. At this point, it is possible to use the owner, group, world protection scheme provided by Unix (1) to protect the directory. In addition a list of login identifiers is created which specifies who may update each element to be processed by SCCS." [2]

Following directory initiation, each document is entered using the admin -n function. Each entry that is made is referred to as an element. As each update is made to a new element, a new generation of that element, known as a delta, is created. The name of each element that is stored in a file by SCCS is preceded by "s.". If a file is added to the directory that does not contain this prefix, it is ignored by the SCCS function calls. When the admin function is called, a number of arguments may be specified that "specify parameters that may affect the file, and may be changed by a subsequent call to admin. The alogin argument is used to create the equivalent of an access control list by listing the login names of users who can apply the delta function to the element, thus creating either a new generation

(1) UNIX is a registered trademark of AT&T Bell Laboratories

(delta) or variant branch."[2]

The initial release, or initial delta, of each code module is entered into the SCCS directory through the admin -n function, thus creating the Master Library. The programmer may update each module in the Master Library by using the get -e function "which indicates that the module will be edited and then the completed document will be reentered into the directory using the delta function. As long as the module being edited was extracted from the SCCS directory using get -e, it can be returned to the library using delta, and all necessary update information will be entered with it. The get function can be used to extract a copy of any document, but after it is edited it cannot be reentered into the library."[2]

"SCCS provides the capability to specify a software build by the way it assigns an SCCS Identification Number (SID) to each output of the delta function."[2] One can get any version of a text or source code by specifying the appropriate SID. "There are straightforward rules regarding how to specify the particular SID desired when get is called. If no SID is specified, the latest release and level is provided." The SID of the resulting call to delta is affected by the SID used when get -e is called.[2]

"The function prs allows for configuration accounting, since it extracts information from the s. files in the SCCS directory and prints them out for the user. Prs can be used to quickly create reports, listing one or two important values such as the last modified date for many SCCS files, or many values for one or two file. Larger reports can also be processed and created using an editor."[2]

A.2 VAX DEC/CMS

"VAX DEC/CMS [7] is also used to track a history of each text file stored in a CMS directory, but CMS does significantly more auditing and cross-checking than admin does. For example, if an editor is used directly to modify a file in a CMS directory, any further use by CMS of that file generates a warning meassage. Any files entered into a CMS directory by other than the CMS utility will cause CMS itself to issue a warning message when it is invoked for that directory. Otherwise, the process of configuration accounting is similar to SCCS.

The CMS CREATE LIBRARY function causes a directory to be set up, and initial logging to start. The project manager enters each element into the directory by using the CMS CREATE ELEMENT function. One must RESERVE an element of a library to modify it,

and it can only be put back into the library using the REPLACE function. If someone else has RESERVEd an element between the original programmer's RESERVE and REPLACE calls, a warning is issued to both programmers and the occurrence is logged. To get a sample copy of the text, such as a program source, the FETCH function will generate the latest generation or any specified generation of an element, but will not allow an edited copy to be reinserted into the library. The SHOW function can be used to audit the information about each element in the library.

Differences between SCCS and DEC/CMS appear concerning software builds. In Unix (1) a build must be either described in a makefile, or else each element to be used in a build must be retrieved from the SCCS directory using get, placed in another directory, and the makefile then may refer to these source files to create the executable build. In CMS, the process of selecting only a subset of source files, including some which are not the most current, is automated by the use of class and group mechanisms. To explain how this works, one must understand the CMS concepts of generations and variants. Each generation of a file corresponds to a Unix (1) delta. Generations are normally numbered in ascending order. CMS also has the capability of creating a variant development line to any generation by specifying in the REPLACE function a variant name. For example, if one RESERVEs generation 3 of an element, then performs a REPLACE/VARIANT = T, this will create generation 3T1 which may then be developed separately from

generation 3. The first time this is used, the equivalent of an SCCS branch delta is created. Branches themselves can have branches, a capability that SCCS does not have.

A group can be defined within a CMS directory, using the CMS CREATE GROUP, and CMS INSERT ELEMENT functions. A group is composed of all generations, including variant generations, of all elements inserted into the group. Groups can be included within other groups. Groups can be defined with a non-empty intersection so that they have overlapping membership.

The CMS CREATE CLASS function, together with the CMS INSERT GENERATION function, can be used to specify the exact elements of a software build, and the DESCRIPTION file can then refer to the entire class by using the /GENERATION=classname qualifier on either the source or action line of a dependency rule. The makefile required by Unix (1) SCCS can be much more complex when it is required to describe a software build for intermediate testing."[2]

(1) Unix is a registered trade mark of Bell Laboratories

GLOSSARY

Automatic Data Processing (ADP) System - An assembly of computer hardware, firmware, and software configured for the purpose of classifying, sorting, calculating, computing, summarizing, transmitting and receiving, storing, and retrieving data with a minimum of human intervention.[1]

Baseline - A set of critical observations or data used for a comparison or a control. A baseline indicates a cutoff point in the design and development of a configuration item beyond which configuration does not evolve without undergoing strict configuration control policies and procedures.

Configuration Accounting - The recording and reporting of configuration item descriptions and all departures from the baseline during design and production.[2]

Configuration Audit - An independent review of computer software for the purpose of assessing compliance with established requirements, standards, and baselines.[2]

Configuration Control - The process of controlling modifications to the system's design, hardware, firmware, software, and documentation which provides sufficient assurance the system is protected against the introduction of improper modification prior to, during, and after system implementation.

Configuration Control Board (CCB) - An established committee that is the final authority on all proposed changes to the ADP system.

Configuration Identification - The identifying of the system configuration throughout the design, development, test, and production tasks.

Configuration Item - The smallest component of hardware, software, firmware, documentation, or any of its discrete portions, which is tracked by the configuration management system.

Configuration Management - The management of changes made to a system's hardware, software, firmware, documentation, tests, test fixtures, and test documentation throughout the development and operational life of the system.

Descriptive Top-Level Specification (DTLS) - A top-level specification that is written in a natural language (e.g., English), an informal program design notation, or a combination of the two.[1]

Firmware - Equipments or devices within which computer programming instructions necessary to the performance of the device's discrete functions are electrically embedded in such a manner that they cannot be electrically altered during normal device operations.[3]

Formal Security Policy Model - An accurate and precise description, in a formal, mathematical language, of the security policy supported by the system.

Formal Top-Level Specification - A top-level specification that is written in a formal mathematical language to allow theorems showing the correspondence of the system specifications to its formal requirements to be hypothesized and formally proven.[1]

Granularity - The relative fineness or courseness by which a mechanism can be adjusted. The phrase "the granularity of a single user" means the access control mechanism can be adjusted to include or exclude any single user.[1]

Hardware - The electric, electronic, and mechanical equipment used for processing data.[3]

Informal Security Policy Model - An accurate and precise description, in a natural language (e.g., English), of the security policy supported by the system.

Software - Various programming aids that are frequently supplied by the manufacturers to facilitate the purchaser's efficient operation of the equipment. Such software items include various assemblers, generators, subroutine libraries, compilers, operating systems, and industry application programs.[6]
Tools - The means for achieving an end result. The tools referred to in this guideline are documentation, procedures, and the organizational body, i.e., the CCB, which all contribute to achieving the control objective of configuration management.

Trusted Computing Base (TCB) - The totality of protection mechanisms within a computer system -- including hardware, firmware, and software -- the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a TCB to correctly enforce a security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (e.g., a user's clearance) related to the security policy.[1]

REFERENCES

1. National Computer Security Center, DOD Trusted Computer System Evaluation Criteria, DOD, DOD 5200.28-STD, 1985.

2. Brown, R. Leonard, "Configuration Management for Development of a Secure Computer System", ATR-88(3777-12)-1, The Aerospace Corporation, 1987.

3. Subcommittee on Automated Information System Security, Working Group #3, "Dictionary of Computer Security Terminology", 23 November 1986.

4. Bersoff, Edward H., Henderson, Vilas D., Siegal, Stanley G., Software Configuration Management, Prentice Hall, Inc., 1980.

5. Samaras, Thomas T., Czerwinski, Frank L., Fundamentals of Configuration Management, Wiley-Interscience, 1971.

6. Sipple, Charles J., Computer Dictionary, Fourth Edition, Howard W. Sams & Co., 1985.

7. Digital Equipment Corporation, VAX DEC/CMS Reference Manual, AA-L372B-TE, Digital Equipment Corporation, 1984.

 FOREWORD This publication, "A Guide to Understanding Audit in Trusted Systems," is being issued by the National Computer Security Center (NCSC) under the authority of and in accordance with Department of Defense (DoD) Directive 5215.1. The guidelines described in this document provide a set of good practices related to the use of auditing in automatic data processing systems employed for processing classified and other sensitive information. Recommendations for revision to this guideline are encouraged and will be reviewed biannually by the National Computer Security Center through a formal review process. Address all proposals for revision through appropriate channels to: National Computer Security Center 9800 Savage Road Fort George G. Meade, MD 20755-6000 Attention: Chief, Computer Security Technical Guidelines _____ Patrick R. Gallagher, Jr. 28 July 1987 Director National Computer Security Center ACKNOWLEDGEMENTS Special recognition is extended to James N. Menendez, National Computer Security Center (NCSC), as project manager of the preparation and production of this document. Acknowledgement is also given to the NCSC Product Evaluations Team who provided the technical guidance that helped form this document and to those members of the computer security community who contributed their time and expertise by actively participating in the review of this document. CONTENTS FOREWORD ..... i ACKNOWLEDGEMENTS ii

PREFACE Throughout this guideline there will be recommendations made that are not included in the Trusted Computer System Evaluation Criteria (the Criteria) as requirements. Any recommendations that are not in the Criteria will be prefaced by the word "should," whereas all requirements will be prefaced by the word "shall." It is hoped that this will help to avoid any confusion.

1. INTRODUCTION 1.1 History of the National Computer Security Center The DoD Computer Security Center (DoDCSC) was established in January 1981 for the purpose of expanding on the work started by the DoD Security Initiative. Accordingly, the Director, National Computer Security Center, has the responsibility for establishing and publishing standards and guidelines for all areas of computer security. In 1985, DoDCSC's name was changed to the National Computer Security Center to reflect its responsibility for computer security throughout the federal government. 1.2 Goal of the National Computer Security Center The main goal of the National Computer Security Center is to encourage the widespread availability of trusted computer systems. In support of that goal a metric was created, the DoD Trusted Computer System Evaluation Criteria (the Criteria), against which computer systems could be evaluated for security. The Criteria was originally published on 15 August 1983 as CSC- STD-001-83. In December 1985 the DoD adopted it, with a few changes, as a DoD Standard, DoD 5200.28-STD. DoD Directive 5200.28, "Security Requirements for Automatic Data Processing (ADP) Systems" has been written to, among other things, require the Department of Defense Trusted Computer System Evaluation Criteria to be used throughout the DoD. The Criteria is the standard used for evaluating the effectiveness of security controls built into ADP systems. The Criteria is divided into four divisions: D, C, B, and A, ordered in a hierarchical manner with the highest division (A) being reserved for systems providing the best available level of assurance. Within divisions C and B there are a number of subdivisions known as classes, which are also ordered in a hierarchical manner to represent different levels of security in these classes. 2. PURPOSE For Criteria classes C2 through A1 the Criteria requires that a user's actions be open to scrutiny by means of an audit. The audit process of a secure system is the process of recording, examining, and reviewing any or all security-relevant activities on the system. This guideline is intended to discuss issues involved in implementing and evaluating an audit mechanism. The purpose of this document is twofold. It provides guidance to manufacturers on how to design and incorporate an effective audit mechanism into their system, and it provides guidance to implementors on how to make effective use of the audit

capabilities provided by trusted systems. This document contains suggestions as to what information should be recorded on the audit trail, how the audit should be conducted, and what protective measures should be accorded to the audit resources. Any examples in this document are not to be construed as the only implementations that will satisfy the Criteria requirement. The examples are merely suggestions of appropriate implementations. The recommendations in this document are also not to be construed as supplementary requirements to the Criteria. The Criteria is the only metric against which systems are to be evaluated. This guideline is part of an on-going program to provide helpful guidance on Criteria issues and the features they address. 3. SCOPE An important security feature of Criteria classes C2 through A1 is the ability of the ADP system to audit any or all of the activities on the system. This guideline will discuss auditing and the features of audit facilities as they apply to computer systems and products that are being built with the intention of meeting the requirements of the Criteria. 4. CONTROL OBJECTIVES The Trusted Computer System Evaluation Criteria gives the following as the Accountability Control

Objective: "Systems that are used to process or handle classified or other sensitive information must assure individual accountability whenever either a mandatory or discretionary security policy is invoked. Furthermore, to assure accountability the capability must exist for an authorized and competent agent to access and evaluate accountability information by a secure means, within a reasonable amount of time and without undue difficulty."(1) The Accountability Control Objective as it relates to auditing leads to the following control objective for auditing: "A trusted computer system must provide authorized personnel with the ability to audit any action that can potentially cause access to, generation of, or effect the release of classified or sensitive information. The audit data will be selectively acquired based on the auditing needs of a particular installation and/or application. However, there must be sufficient granularity in the audit data to support tracing the auditable events to a specific individual (or process) who has taken the actions or on whose behalf the actions were taken."(1) 3 5. OVERVIEW OF AUDITING PRINCIPLES Audit trails are used to detect and deter penetration of a computer system and to reveal usage that identifies misuse. At the discretion of the auditor, audit trails may be limited to specific events or may encompass all of the activities on a system. Although not required by the TCSEC, it should be possible for the target of the audit mechanism to be either a subject or an object. That is to say, the audit mechanism should be capable of monitoring every time John accessed the system as well as every time the nuclear reactor file was accessed; and likewise every time John accessed the nuclear reactor file. 5.1 Purpose of the Audit Mechanism The audit mechanism of a computer system has five important security goals. First, the audit mechanism must "allow the review of patterns of access to individual objects, access histories of specific processes and individuals, and the use of the various protection mechanisms supported by the system and their effectiveness."(2) Second, the audit mechanism must allow discovery of both users' and outsiders' repeated attempts to bypass the protection mechanisms. Third, the audit mechanism must allow discovery of any use of privileges that may occur when a user assumes a functionality with privileges greater than his or her own, i.e., programmer to administrator. In this case there may be no bypass of security controls but nevertheless a violation is made possible. Fourth, the audit mechanism must act as a deterrent against perpetrators' habitual attempts to bypass the system protection mechanisms. However, to act as a deterrent, the perpetrator must be aware of the audit mechanism's existence and its active use to detect any attempts to bypass system protection mechanisms. The fifth goal of the audit mechanism is to supply "an additional form of user assurance that attempts to bypass the protection mechanisms are recorded and discovered."(2) Even if the attempt to bypass the protection mechanism is successful, the audit trail will still provide assurance by its ability to aid in assessing the damage done by the violation, thus improving the system's ability to control the damage. 5.2. Users of the Audit Mechanism "The users of the audit mechanism can be divided into two groups. The first group consists of the auditor, who is an individual with administrative duties, who selects the events to be audited on the system, sets up the audit flags which enable the recording

of those events, and analyzes the trail of audit events."(2) In some systems the duties of the auditor may be encompassed in the duties of the system security administrator. Also, at the lower classes, the auditor role may be performed by the system administrator. This document will refer to the person responsible for auditing as the system security administrator, although it is understood that the auditing guidelines may apply to system administrators and/or system security administrators and/or a separate auditor in some ADP systems. "The second group of users of the audit mechanism consists of the system users themselves; this group includes the administrators, the operators, the system programmers, and all other users. They are considered users of the audit mechanism not only because they, and their programs, generate audit events,"(2) but because they must understand that the audit mechanism exists and what impact it has on them. This is important because otherwise the user deterrence and user assurance goals of the audit mechanism cannot be achieved. 5.3 Aspects of Effective Auditing 5.3.1. Identification/Authentication Logging in on a system normally requires that a user enter the specified form of identification (e.g., login ID, magnetic strip) and a password (or some other mechanism) for authentication. Whether this information is valid or invalid, the execution of the login procedure is an auditable event and the

identification entered may be considered to be auditable information. It is recommended that authentication information, such as passwords, not be forwarded to the audit trail. In the event that the identification entered is not recognized as being valid, the system should also omit this information from the audit trail. The reason for this is that a user may have entered a password when the system expected a login ID. If the information had been written to the audit trail, it would compromise the password and the security of the user. There are, however, environments where the risk involved in recording invalid identification information is reduced. In systems that support formatted terminals, the likelihood of password entry in the identification field is markedly reduced, hence the recording of identification information would pose no major threat. The benefit of recording the identification information is that break-in attempts would be easier to detect and identifying the perpetrator would also be assisted. The

information gathered here may be necessary for any legal prosecution that may follow a security violation. 5.3.2 Administrative All systems rated at class C2 or higher shall have audit capabilities and personnel designated as responsible for the audit procedures. For the C2 and B1 classes, the duties of the system operators could encompass all functions including those of the auditor. Starting at the B2 class, there is a requirement for the TCB to support separate operator and administrator functions. In addition, at the B3 class and above, there is a requirement to identify the system security administrator functions. When one assumes the system security administrator role on the system, it shall be after taking distinct auditable action, e.g., login procedure. When one with the privilege of assuming the role is on the system, the act of assuming that role shall also be an auditable event. 5.3.3 System Design The system design should include a mechanism to invoke the audit function at the request of the system security administrator. A mechanism should also be included to determine if the event is to be selected for inclusion as an audit trail entry. If pre-selection of events is not implemented, then all auditable events should be forwarded to the audit trail. The Criteria requirement for the administrator to be able to select events based on user identity and/or object security classification must still be able to be satisfied. This requirement can be met by allowing post-selection of events through the use of queries. Whatever reduction tool is used to analyze the audit trail shall be provided by the vendor. 5.4 Security of the Audit Audit trail software, as well as the audit trail itself, should be protected by the Trusted Computing Base and should be subject to strict access controls. The security requirements of the audit mechanism are the following: (1) The event recording mechanism shall be part of the TCB and shall be protected from unauthorized modification or circumvention. (2) The audit trail itself shall be protected by the TCB from

unauthorized access (i.e., only the audit personnel may access the audit trail). The audit trail shall also be protected from unauthorized modification. (3) The audit-event enabling/disabling mechanism shall be part of the TCB and shall remain inaccessible to the unauthorized users.(2) At a minimum, the data on the audit trail should be considered to be sensitive, and the audit trail itself shall be considered to be as sensitive as the most sensitive data contained in the system. When the medium containing the audit trail is physically removed from the ADP system, the medium should be accorded the physical protection required for the highest sensitivity level of data contained in the system. 7

6. MEETING THE CRITERIA REQUIREMENTS This section of the guideline will discuss the audit requirements in the Criteria and will present a number of additional recommendations. There are four levels of audit requirements. The first level is at the C2 Criteria class and the requirements continue evolving through the B3 Criteria class. At each of these levels, the guideline will list some of the events which should be auditable, what information should be on the audit trail, and on what basis events may be selected to be audited. All of the requirements will be prefaced by the word "shall," and any additional

recommendations will be prefaced by the word "should." 6.1 The C2 Audit Requirement 6.1.1 Auditable Events The following events shall be subject to audit at the C2 class: * Use of identification and authentication mechanisms * Introduction of objects into a user's address space * Deletion of objects from a user's address space * Actions taken by computer operators and system administrators and/or system security administrators * All security-relevant events (as defined in Section 5 of this guideline)

* Production of printed output 6.1.2 Auditable Information The following information shall be recorded on the audit trail at the C2 class: * Date and time of the event * The unique identifier on whose behalf the subject generating the event was operating * Type of event * Success or failure of the event

* Origin of the request (e.g., terminal ID) for identification/authentication events * Name of object introduced, accessed, or deleted from a user's address space * Description of modifications made by the system administrator to the user/system security databases 6.1.3 Audit Basis At the C2 level, the ADP System Administrator shall be able to audit based on individual identity. The ADP System Administrator should also be able to audit based on object identity. 6.2 The B1 Audit Requirement 6.2.1 Auditable Events The Criteria specifically adds the following to the list of events that shall be auditable at the B1 class: * Any override of human readable output markings (including overwrite of sensitivity label markings and the turning off of labelling capabilities) on paged, hard-copy output devices * Change of designation (single-level to/from multi-level) of any communication channel or I/O device * Change of sensitivity level(s) associated with a single-level communication channel or I/O device * Change of range designation of any multi-level communication channel or I/O device 6.2.2 Auditable Information The Criteria specifically adds the following to the list of information that shall be recorded on the audit trail at the B1 class: * Security level of the object

The following information should also be recorded on the audit trail at the B1 class: * Subject sensitivity level 6.2.3 Audit Basis In addition to previous selection criteria, at the B1 level the Criteria specifically requires that the ADP System Administrator shall be able to audit based on individual identity and/or object security level. 6.3 The B2 Audit Requirement 6.3.1 Auditable Events The Criteria specifically adds the following to the list of events that shall be auditable at the B2 class: * Events that may exercise covert storage channels 6.3.2 Auditable Information No new requirements have been added at the B2 class. 6.3.3 Audit Basis In addition to previous selection criteria, at the B2 level the Criteria specifically requires that "the TCB shall be able to audit the identified events that may be used in the exploitation of covert storage channels." The Trusted Computing Base shall audit covert storage channels that exceed ten bits per second.(1)

The Trusted Computing Base should also provide the capability to audit the use of covert storage mechanisms with bandwidths that may exceed a rate of one bit in ten seconds. 6.4 The B3 Audit Requirement 6.4.1 Auditable Events The Criteria specifically adds the following to the list of events that shall be auditable at the B3 class: * Events that may indicate an imminent violation of the

system's security policy (e.g., exercise covert timing channels) 6.4.2 Auditable Information No new requirements have been added at the B3 class. 6.4.3 Audit Basis In addition to previous selection criteria, at the B3 level the Criteria specifically requires that "the TCB shall contain a mechanism that is able to monitor the occurrence or accumulation of security auditable events that may indicate an imminent

violation of security policy. This mechanism shall be able to immediately notify the system security administrator when thresholds are exceeded and, if the occurrence or accumulation of these security-relevant events continues, the system shall take the least disruptive action to terminate the event."(1) Events that would indicate an imminent security violation would include events that utilize covert timing channels that may exceed a rate of ten bits per second and any repeated unsuccessful login attempts. Being able to immediately notify the system security administrator when thresholds are exceeded means that the mechanism shall be able to recognize, report, and respond to a violation of the security policy more rapidly than required at lower levels of the Criteria, which usually only requires the System Security Administrator to review an audit trail at some time after the event. Notification of the violation "should be at the same priority as any other TCB message to an operator."(5) "If the occurrence or accumulation of these security-relevant events continues, the system shall take the least disruptive action to terminate the event."(1) These actions may include locking the terminal of the user who is causing the event or terminating the suspect's process(es). In general, the least disruptive action is application dependent and there is no requirement to demonstrate that the action is the least disruptive of all possible actions. Any action which terminates the event is acceptable, but halting the system should be the last resort.

7.5 The A1 Audit Requirement 7.5.1 Auditable Events No new requirements have been added at the A1 class. 7.5.2 Auditable Information No new requirements have been added at the A1 class. 7.5.3 Audit Basis No new requirements have been added at the A1 class.

7. POSSIBLE IMPLEMENTATION METHODS The techniques for implementing the audit requirements will vary from system to system depending upon the characteristics of the software, firmware, and hardware involved and any optional features that are to be available. Technologically advanced techniques that are available should be used to the best advantage in the system design to provide the requisite security as well as cost-effectiveness and performance. 7.1 Pre/Post Selection of Auditable Events There is a requirement at classes C2 and above that all security- relevant events be

auditable. However, these events may or may not always be recorded on the audit trail. Options that may be exercised in selecting which events should be audited include a pre-selection feature and a post-selection feature. A system may choose to implement both options, a pre-selection option only, or a post-selection option only. If a system developer chooses not to implement a general pre/post selection option, there is still a requirement to allow the administrator to selectively audit the actions of specified users for all Criteria classes. Starting at the B1 class, the administrator shall also be able to audit based on object security level. There should be options to allow selection by either individuals or groups of users. For example, the administrator may select events related to a specified individual or select events related to individuals included in a specified group. Also, the administrator may specify that events related to the audit file be selected or, at classes B1 and above, that accesses to objects with a given sensitivity level, such as Top Secret, be selected. 7.1.1 Pre-Selection For each auditable event the TCB should contain a mechanism to indicate if the event is to be recorded on the audit trail. The system security administrator or designee shall be the only person authorized to select the events to be recorded. Pre-selection may be by user(s) identity, and at the B1 class and above, pre-selection may also be possible by object security level. Although the system security administrator shall be authorized to select which events are to be recorded, the system security administrator should not be able to exclude himself from being audited. 13

Although it would not be recommended, the system security administrator may have the capability to select that no events be recorded regardless of the Criteria requirements. The intention here is to provide flexibility. The purpose of designing audit features into a system is not to impose the Criteria on users that may not want it, but merely to provide the capability to implement the requirements. A disadvantage of pre-selection is that it is very hard to predict what events may be of security-relevant interest at a future date. There is always the possibility that events not pre-selected could one day become security-relevant, and the potential loss from not auditing these events would be impossible to determine. The advantage of pre-selection could possibly be better performance as a result of not auditing all the events on the system. 7.1.2 Post-Selection If the post-selection option to select only specified events from an existing audit trail is implemented, again, only authorized personnel shall be able to make this selection. Inclusion of this option requires that the system should have trusted facilities (as described in section 9.1) to accept query/retrieval requests, to expand any compressed data, and to output the requested data. The main advantage of post-selection is that information that may prove useful in the future is already recorded on an audit trail and may be queried at any time. The disadvantage involved in post-selection could possibly be degraded performance due to the writing and storing of what could possibly be a very large audit trail. 7.2 Data Compression

"Since a system that selects all events to be audited may generate a large amount of data, it may be necessary to encode the data to conserve space and minimize the processor time required" to record the audit records.(3) If the audit trail is encoded, a complementary mechanism must be included to decode the data when required. The decoding of the audit trail may be done as a preprocess before the audit records are accessed by the database or as a postprocess after a relevant record has been

found. Such decoding is necessary to present the data in an understandable form both at the administrators terminal and on batch reports. The cost of compressing the audit trail would be the time required for the compression and expansion processes. The benefit of compressing data is the savings in storage and the savings in time to write the records to the audit trail. 7.3 Multiple Audit Trails All events included on the audit trail may be written as part of the same audit trail, but some systems may prefer to have several distinct audit trails, e.g., one would be for "user" events, one for "operator" events, and one for "system security administrator" events. This would result in several smaller trails for subsequent analysis. In some cases, however, it may be necessary to combine the information from the trails when questionable events

occur in order to obtain a composite of the sequence of events as they occurred. In cases where there are multiple audit trails, it is preferred that there be some accurate, or at least synchronized, time stamps across the multiple logs. Although the preference for several distinct audit trails may be present, it is important to note that it is often more useful that the TCB be able to present all audit data as one comprehensive audit trail. 7.4 Physical Storage A factor to consider in the selection of the medium to be used for the audit trail would be the expected usage of the system. The I/O volume for a system with few users executing few applications would be quite different from that of a large system with a multitude of users performing a variety of applications. In any case, however, the system should notify the system operator or administrator when the audit trail medium is approaching its storage capacity. Adequate advance notification to the operator is especially necessary if human intervention is required. If the audit trail storage medium is saturated before it is replaced, the operating system shall detect this and take some appropriate action such as: 1. Notifying the operator that the medium is "full" and action is necessary. The system should then stop and require rebooting. Although a valid option, this action creates a

severe threat of denial-of-service attacks. 2. Storing the current audit records on a temporary medium with the intention of later migration to the normal operational medium, thus allowing auditing to continue. This temporary storage medium should be afforded the same protection as the regular audit storage medium in order to prevent any attempts to tamper with it. 3. Delaying input of new actions and/or slowing down current operations to prevent any action that requires use of the audit mechanism. 4. Stopping until the administrative personnel make more space available for writing audit records. 5. Stopping auditing entirely as a result of a decision by the system security administrator. Any action that is taken in response to storage overflow shall be audited. There is, however, a case in which the action taken may not be audited that deserves mention. It is possible to have the system security administrator's decisions embedded in the system logic. Such pre-programmed choices, embedded in the system logic, may be triggered automatically and this action may not be audited. Still another consideration is the speed at which the medium operates. It should be able to accommodate the "worst case" condition such as when there are a large number of users on the system and all auditable events are to be recorded. This worst case rate should be estimated during the system design phase and (when possible) suitable hardware should be selected for this purpose. Regardless of how the system handles audit trail overflow, there must be a way to archive all of the audit data. 7.5 Write-Once Device For the lower Criteria classes (e.g., C2, B1) the audit trail may be the major tool used in detecting security compromises. Implicit in this is that the audit resources should provide the maximum protection possible. One technique that may be employed to protect the audit trail is to record it on a mechanism designed to be a write-only device. Other choices would be to set the designated device to write-once mode by disabling the

read mechanism. This method could prevent an attacker from erasing or modifying the data already written on the audit trail because the attacker will not be able to go back and read or find the data that he or she wishes to modify.

If a hardware device is available that permits only the writing of data on a medium, modification of data already recorded would be quite difficult. Spurious messages could be written, but to locate and modify an already recorded message would be difficult. Use of a write-once device does not prevent a penetrator from modifying audit resources in memory, including any buffers, in the current audit trail. If a write-once device is used to record the audit trail, the medium can later be switched to a compatible read device to allow authorized personnel to analyze the information on the audit trail in order to detect any attempts to penetrate the system. If a penetrator modified the audit software to prevent writing records on the audit

trail, the absence of data during an extended period of time would indicate a possible security compromise. The disadvantage of using a write-once device is that it necessitates a delay before the audit trail is available for analysis by the administrator. This may be offset by allowing the system security administrator to review the audit trail in real-time by getting copies of all audit records on their way to the device. 7.6 Forwarding Audit Data If the facilities are available, another method of protecting the audit trail would be to forward it to a dedicated processor. The audit trail should then be more readily available for analysis by the system security administrator. 17

8. OTHER TOPICS 8.1 Audit Data Reduction Depending upon the amount of activity on a system and the audit selection process used, the audit trail size may vary. It is a safe assumption though, that the audit trail would grow to sizes that would necessitate some form of audit data reduction. The data reduction tool would most likely be a batch program that would interface to the system security administrator. This batch run could be a combination of database query language and a report generator with the input being a standardized audit file. Although they are not necessarily part of the TCB, the audit reduction tools should be maintained under the same configuration control system as the remainder of the system. 8.2 Availability of Audit Data In standard data processing, audit information is recorded as it occurs. Although most information is not required to be immediately available for real-time analysis, the system security administrator should have the capability to retreive audit information within minutes of its recording. The delay between recording audit information and making it available for analysis should be minimal, in the range of several minutes. For events which do require immediate attention, at the B3 class and above, an alert shall be sent out to the system security administrator. In systems that store the audit trail in a buffer, the system security administrator should have the capability to cause the buffer to be written out. Regarding real-time alarms, where they are sent is system dependent. 8.3 Audit Data Retention The exact period of time required for retaining the audit trail is site dependent and should be documented in the site's operating procedures manual. When trying to arrive at the optimum time for audit trail retention, any time restrictions on the storage medium should be considered. The storage medium used must be able to reliably retain the audit data for the amount of time required by the site. The audit trail should be reviewed at least once a week. It is very possible that once a week may be too long to wait to review

the audit trail. Depending on the amount of audit data expected by the system, this parameter should be adjusted accordingly. The recommended time in between audit trail reviews should be documented in the Trusted Facility Manual. 8.4 Testing The audit resources, along with all other resources protected by the TCB, have increasing assurance requirements at each higher Criteria class. For the lower classes, an audit trail would be a major factor in detecting penetration attempts. Unfortunately, at these lower classes, the audit resources are more susceptible to penetration and corruption. "The TCB must provide some assurance that the data will still be there when the administrator tries to use it."(3) The testing requirement recognizes the vulnerability of the audit trail, and starting with the C2 class, shall include a search for obvious flaws that would corrupt or destroy the audit trail. If the audit trail is corrupted or destroyed, the existence of such flaws indicates that the system can be penetrated. Testing should also be performed to uncover any ways of circumventing the audit mechanisms. The "flaws found in testing may be neutralized in any of a number of ways. One way available to the system designer is to audit all uses of the mechanism in which the flaw is found and to log such events."(3) An attempt should be made to remove the flaw. At class B2 and above, it is required that all detected flaws shall be corrected or else a lower rating will be given. If during testing the audit trail appears valid, analysis of this data can verify that it does or does not accurately reflect the events that should be included on the audit trail. Even though system assurances may increase at the higher classes, the audit trail is still an effective tool during the testing phase as well as operationally in detecting actual or potential security compromises. 8.5 Documentation Starting at the C2 class, documentation concerning the audit requirements shall be contained in the Trusted Facility Manual. The Trusted Facility Manual shall explain the procedures to

record, examine, and maintain audit files. It shall detail the audit record structure for each type of audit event, and should include what each field is and what the size of the field is. The Trusted Facility Manual shall also include a complete

description of the audit mechanism interface, how it should be used, its default settings, cautions about the trade-offs involved in using various configurations and capabilities, and how to set up and run the system such that the audit data is afforded appropriate protection. If audit events can be pre- or post-selected, the manual should also describe the tools and mechanisms available and how they are to be used. 8.6 Unavoidable Security Risks There are certain risks contained in the audit process that exist simply because there is no way to prevent these events from ever occurring. Because there are certain unpredictable factors involved in auditing, i.e., man, nature, etc., the audit mechanism may never be one hundred per cent reliable. Preventive measures may be taken to minimize the likelihood of any of these factors adversely affecting the security provided by the audit mechanism, but no audit mechanism will ever be risk free. 8.6.1 Auditing Administrators/Insider Threat Even with auditing mechanisms in place to detect and deter security violations, the threat of the perpetrator actually being the system security administrator or someone involved with the system security design will always be present. It is quite possible that the system security administrator of a secure system could stop the auditing of activities while entering the system and corrupting files for personal benefit. These authorized personnel, who may also have access to identification and authentication information, could also choose to enter the system disguised as another user in order to commit crimes under a false identity. Management should be aware of this risk and should be certain to exercise discretion when selecting the system security administrator. The person who is to be selected for a trusted position, such as the system security administrator, should be subject to a background check before being granted the privileges that could one day be used against the employer. The system security administrator could also be watched to ensure that there are no unexplained variances in normal duties. Any deviation from the norm of operations may indicate that a violation of security has occurred or is about to occur.

An additional security measure to control this insider threat is to ensure that the system administrator and the person responsible for the audit are two different people. "The separation of the auditor's functions, databases, and access privileges from those of the system administrator is an important application of the separation of privilege and least privilege principles. Should such a separation not be performed, and should the administrator be allowed to undertake auditor functions or vice-versa, the entire security function would become the responsibility of a single, unaccountable individual."(2) Another alternative may be to employ separate auditor roles. Such a situation may give one person the authority to turn off the audit mechanism, while another person may have the authority to turn it back on. In this case no individual would be able to turn off the audit mechanism, compromise the system, and then turn it back on. 8.6.2 Data Loss Although the audit software and hardware are reliable security mechanisms, they are not infallible. They, like the rest of the system, are dependent upon constant supplies of power and are readily subject to interruption due to mechanical or power failures. Their failure can cause the loss or destruction of valuable audit data. The system security administrator should be aware of this risk and should establish some procedure that would ensure that the audit trail is preserved somewhere. The system security administrator should duplicate the audit trail on a removable medium at certain points in time to minimize the data loss in the event of a system failure. The Trusted Facility Manual should include what the possibilities and nature of loss exposure are, and how the data may be recovered in the event that a catastrophe does occur. If a mechanical or power failure occurs, the system security administrator should ensure that audit mechanisms still function properly after system recovery. For example, any auditing

mechanism options pre-selected before the system malfunction must still be the ones in operation after the system recovery. 21 9. AUDIT SUMMARY For classes C2 and above, it is required that the TCB "be able to create, maintain, and protect from modification or unauthorized access or destruction an audit trail of accesses to the objects it protects."(1) The audit trail plays a key role in performing damage assessment in the case of a corrupted system. The audit trail shall keep track of all security-relevant events such as the use of identification and authentication mechanisms, introduction of objects into a user's address space, deletion of objects from the system, system administrator actions, and any other events that attempt to violate the security policy of the system. The option should exist that either all activities be audited or that the system security administrator select the events to be audited. If it is decided that all activities should be audited, there are overhead factors to be considered. The storage space needed for a total audit would generally require more operator maintenance to prevent any loss of data and to provide adequate protection. A requirement exists that authorized personnel shall be able to read all events recorded on the audit trail. Analysis of the total audit trail would be both a difficult and time-consuming task for the administrator. Thus, a selection option is required which may be either a pre-selection or post-selection option. The audit trail information should be sufficient to reconstruct a complete sequence of security-relevant events and processes for a system. To do this, the audit trail shall contain the following information: date and time of the event, user, type of event, success or failure of the event, the origin of the request, the name of the object introduced into the user's address space, accessed, or deleted from the storage system, and at the B1 class and above, the sensitivity determination of the object. It should be remembered that the audit trail shall be included in the Trusted Computing Base and shall be accorded the same protection as the TCB. The audit trail shall be subject to strict access controls. An effective audit trail is necessary in order to detect and evaluate hostile attacks on a system.

GLOSSARY

Administrator - Any one of a group of personnel assigned to supervise all or a portion of an ADP system. Archive - To file or store records off-line. Audit - To conduct the independent review and examination of system records and activities. Auditor - An authorized individual with administrative duties, whose duties include selecting the events to be audited on the system, setting up the audit flags which enable the recording of those events, and analyzing the trail of audit events.(2) Audit Mechanism - The device used to collect, review, and/or examine system activities. Audit Trail - A set of records that collectively provide documentary evidence of processing used to aid in tracing from original transactions forward to related records and reports, and/or backwards from records and reports to their component source transactions.(1) Auditable Event - Any event that can be selected for inclusion in the audit trail. These events should include, in addition to security-relevant events, events taken to recover the system after failure and any events that might prove to be security-relevant at a later time. Authenticated User - A user who has accessed an ADP system with a valid identifier and authentication combination. Automatic Data Processing (ADP) System - An assembly of computer hardware, firmware, and software configured for the purpose of classifying, sorting, calculating, computing, summarizing, transmitting and receiving, storing, and retrieving data with a minimum of human intervention.(1) Category - A grouping of classified or unclassified sensitive information, to which an additional restrictive label is applied (e.g., proprietary, compartmented information) to signify that personnel are granted access to the information only if they have formal approval or other appropriate authorization.(4) Covert Channel - A communication channel that allows a process to transfer information in a manner that violates the system's security policy.(1)

Covert Storage Channel - A covert channel that involves the direct or indirect writing of a storage location by one process and the direct or indirect reading of the storage location by another process. Covert storage channels typically involve a finite resource (e.g., sectors on a disk) that is shared by two subjects at

different security levels.(1) Covert Timing Channel - A covert channel in which one process signals information to another by modulating its own use of system resources (e.g., CPU time) in such a way that this manipulation affects the real response time observed by the second process.(1) Flaw - An error of commission, omission or oversight in a system that allows protection mechanisms to be bypassed.(1) Object - A passive entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are: records, blocks, pages, segments, files, directories, directory trees and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes, etc.(1) Post-Selection - Selection, by authorized personnel, of specified events that had been recorded on the audit trail. Pre-Selection - Selection, by authorized personnel, of the auditable events that are to be recorded on the audit trail. Security Level - The combination of a hierarchical classification and a set of non-hierarchical categories that represents the sensitivity of information.(1) Security Policy - The set of laws, rules, and practices that regulate how an organization manages, protects, and distributes sensitive information.(1) Security-Relevant Event - Any event that attempts to change the security state of the system, (e.g., change discretionary access controls, change the security level of the subject, change user password, etc.). Also, any event that attempts to violate the security policy of the system, (e.g., too many attempts to login, attempts to violate the mandatory access control limits of a device, attempts to downgrade a file, etc.).(1) Sensitive Information - Information that, as determined by a competent authority, must be protected because its unauthorized disclosure, alteration, loss, or destruction will at least cause perceivable damage to someone or something.(1) 24

Subject - An active entity, generally in the form of a person, process, or device that causes information to flow among objects or changes the system state. Technically, a process/domain pair.(1) Subject Sensitivity Level - The sensitivity level of the objects to which the subject has both read and write access. A subject's sensitivity level must always be less than or equal to the clearance of the user the subject is associated with.(4) System Security Administrator - The person responsible for the security of an Automated Information System and having the authority to enforce the security safeguards on all others who have access to the Automated Information System.(4) Trusted Computing Base (TCB) - The totality of protection mechanisms within a computer system -- including hardware, firmware, and software -- the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a TCB to correctly enforce a security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (e.g., a user's clearance) related to the security policy.(1) User - Any person who interacts directly with a computer system.(1)

REFERENCES 1. National Computer Security Center, DoD Trusted Computer System Evaluation Criteria, DoD, DoD 5200.28-STD, 1985. 2. Gligor, Virgil D., "Guidelines for Trusted Facility Management and Audit," University of Maryland, 1985. 3. Brown, Leonard R., "Guidelines for Audit Log Mechanisms in Secure Computer Systems," Technical Report TR-0086A(2770-29)-1, The Aerospace Corporation, 1986. 4. Subcommittee on Automated Information System Security, Working Group #3, "Dictionary of Computer Security Terminology," 23 November 1986. 5. National Computer Security Center, Criterion Interpretation, Report No. C1-C1-02-87, 1987.