A Guide to Understanding Design Documentation

ACKNOWLEDGMENTS

Special recognition is extended to James N. Menendez, National Computer Security Center (NCSC), as project manager and co-author of this document. Recognition is also extended to Don Brinkley and Rick Dedrick, Unisys, as co-authors of this document.

Special acknowledgment is given to Barbara Mayer, NCSC, for her invaluable input and review of this document, which has resulted in its current form.

Acknowledgment 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.

A Guide to Understanding Design Documentation

NCSC-TG-07 Version 1

1. INTRODUCTION

1.1 Purpose

The Trusted Computer System Evaluation Criteria (TCSEC) is the standard used for evaluating the effectiveness of security controls built into Automated Data Processing (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 those 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 trust in these classes.

Design Documentation is a TCSEC requirement for classes C1 and above. The purpose of this guideline is to provide developers of trusted computer systems with guidance in understanding and meeting the design documentation requirements contained in the TCSEC. To accomplish this, the guideline addresses two goals. First, the guideline increases the vendors' awareness of the importance of design documentation to the security of their system throughout the system life-cycle. Second, the guideline forms an initial basis of understanding between the vendor and evaluator communities concerning what is expected by the evaluation team in the review process and deliverables for design documentation.

Any examples in this document are not to be construed as the only implementation that will satisfy the TCSEC requirement. The examples are merely suggested 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 will be evaluated.

This guideline is part of the Technical Guidelines Program to provide helpful guidance on TCSEC issues and the features they address.

1.2 Scope

Design Documentation is a TCSEC requirement for classes C1 through A1. It is one of the four types of documentation required by the TCSEC. The other three documentation requirements are for a Trusted Facility Manual (TFM), Security Features Users Guide (SFUG), and Test Plan Documentation. The role of Design Documentation is to identify and describe the Trusted Computing Base (TCB) and its security features. Only Design Documentation for the TCB is required to meet the TCSEC requirements, but it is strongly recommended that design documentation exist for the entire system. Throughout this document, the word system will be used as the object of design documentation to include the TCB and the untrusted portions of the system. However, it should be emphasized that the TCSEC requirements are based solely on the design documentation of the TCB.

Design Documentation assists vendors during the system life-cycle by thoroughly defining the policies that the system enforces. It also provides the material by which the evaluator can assess whether, and to what degree, the design intent was carried into the implementation. The design documentation is intended to guide the implementation of the product; it is not intended merely as an abstract philosophical exercise completely divorced from the "real" product.

Design documentation also increases the developer's level of understanding of the system. It should facilitate the correct implementation of the intended behavior and features of the system. This guideline will discuss design documentation and its features as they apply to computer systems and products that are being built with the intention of meeting the requirements of the TCSEC.

1.3 Control Objective

Each of the TCSEC requirements serves to ensure that one of the three basic control objectives for trusted computing - security policy, accountability, and assurance - are satisfied. Throughout the system life-cycle, design documentation aids in attaining the third objective, assurance, by helping to "substantiate claims for the completeness of access mediation and degree of tamper resistance." [5]

- 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."[5]

Design documentation plays an important role in providing this life-cycle assurance. It demonstrates that correct implementation and enforcement of the system's security policy exists throughout the system's life-cycle. As it relates to this control objective, design documentation facilitates the efforts of vendors and system developers in modifying and maintaining the system throughout its life-cycle, without compromising the trustworthiness of the system.

In addition, design documentation serves as a useful training tool. Design documentation presents a technical history of the system, containing documentation on past changes to the system as well as the current system. It can be used in the training of new systems programmers and hardware engineers to familiarize them with the system.

2. OVERVIEW OF DESIGN DOCUMENTATION PRINCIPLES

Design documentation is a requirement for TCSEC classes C1 and above. It provides a means of communicating the design of a system to developers that enables them to comprehend the design principles of the system and to make changes or upgrades to the system without compromising the trustworthiness of the system. The information contained in the design documentation provides a rationale as to why a system is designed as it is and whether changes to the system will alter the intent of the design.

Design documentation plays an important role in the life-cycle maintenance of a system and should not be viewed as a burden to system development. This document should help developers understand the importance of design documentation in the life-cycle of computer systems, as well as to the maintenance of trust in these systems. Developers should recognize the importance of meeting the purpose and intent of the TCSEC design documentation requirements as opposed to meeting them in a strictly mechanical fashion.

2.1 Purpose of Design Documentation

The primary purpose of design documentation is to define and describe the properties of a system. As it relates to the TCSEC, design documentation provides an explanation of how the security policy of a system is translated into a technical solution through the TCB hardware, software, and firmware.

Design documentation explains the system's protection mechanisms so that the effect a change may have on the security of the system can be evaluated prior to a change being performed. It relates the TCSEC requirements to the architecture of a system and guides the implementation of the system under development. Complete documentation ensures that the vendor has an understanding of what elements of the system are protection critical. Design documentation explains the system design to the vendor's development team and enables the developers to understand the design of the system well enough to maintain the system and to perform any necessary changes to it without adversely affecting the trustworthiness of the system. In addition, the design documentation assists the evaluators by providing them with a vehicle by which the completeness and correctness of the implementation can be assessed.

2.2 Design Documentation Development for Evaluation

Developers should incorporate the design documentation requirements into the system development process. A plan that addresses each design documentation requirement should be developed early in the development phase and shared with the National Computer Security Center evaluators to ensure the thoroughness of the documentation.

Iterative development of the design documentation is the key to minimizing vendor and evaluator efforts during the evaluation process. Vendors should precede their design documentation with the submittal of an outline of the design documentation to the evaluators. This outline should contain, among other things, a statement of purpose and the intended audience of the design documentation. Then, through a process of draft submittal, evaluator comments and requests for additional information, and draft revision the design documentation requirements will be met. This guideline should expedite this process by bringing the vendor's first drafts closer to evaluator expectations and by facilitating convergence between vendor product and evaluator expectations. If vendors establish a dialogue with evaluators early in the design documentation development and solicit their comments on early and subsequent drafts of the design documentation, both vendors and evaluators will save a great deal of time and effort in completing the evaluation process.

2.3 Level of Detail of Design Documentation

The level of detail of design documentation will determine its usefulness and adequacy in meeting the TCSEC requirements, as well as its usefulness to the vendor in the development and maintenance of the system. For evaluators, the level of detail of the design documentation is reviewed to ensure that the system developer understands the design of the system well enough to make changes or upgrades to the system without compromising the trustworthiness of the system. Design documentation also ensures that the developer understands the overall security concepts that are required to be part of the system design. How well the security properties of the system are documented, and how this information is integrated into the design documentation will determine whether or not the level of detail of the design documentation is sufficient in meeting the TCSEC requirements.

The design documentation shall be detailed enough to serve as a useful tool for vendor maintenance of the system and shall clearly indicate what elements of the design impact the trustworthiness of the system. One purpose behind design documentation is to assist vendors in maintaining the system and should not present a burden to the vendor in terms of quantity or detail. A good rule of thumb is that the level of detail of design documentation should be sufficient to permit an individual with a degree in Computer Science, Electrical Engineering, or the equivalent with knowledge and skills in programming, hardware, or firmware development to understand the system design and be able to design system modifications without adversely affecting trustworthiness of the system.

2.4 Level of Effort for Meeting the Requirements

The level of effort necessary for developing satisfactory design documentation has historically been underestimated because the intent and implications of the TCSEC requirements for design documentation have seldom been completely understood. An important factor to consider when deciding on an appropriate level of effort is the importance of the design documentation throughout the system life-cycle. Well structured design documentation that is carefully planned and developed will make it easier to understand the design of the system.

The level of effort necessary for a vendor to meet the design documentation requirements varies from system to system. The level of effort generally will depend upon the necessary level of detail, which depends upon the class of evaluation and the complexity of the system being evaluated. The requirements for TCSEC classes C1 and C2 may be met by simply following good engineering documentation practices, but as the TCSEC class level increases, so does the level of detail and effort necessary for meeting the TCSEC requirements.

In terms of quantity, the length of design documentation at the higher classes has been found to be roughly comparable in bulk to the source listings of the overall system. In general, producing the design documentation may require several man months to a man year of system development time at Classes C1 and C2, and up to several man years at the higher classes. Although developing design documentation for a system may be time consuming, this time will be amply rewarded by the ease of system maintainability during its life-cycle.

2.5 Format of Design Documentation

The format and style for each vendor's design documentation is specific to that vendor, and to suggest a specific format would restrict vendors in developing their design documentation. Although this guideline addresses distinct requirements for design documentation, it should not be assumed that separate documents are necessary to meet each requirement. Indeed, the design documentation shall address each of the requirements, but it is acceptable for evaluators to be pointed to a number of documents to address a specific requirement. Also, graphics serve as a useful adjunct to design documentation, although not sufficient alone to meet the TCSEC requirement for design documentation. Developers may choose to use graphics to describe a system in addition to other design documentation.

Differences among computer system architectures, designs, and implementation approaches make developing a standard format for design documentation inadvisable. In addition, the format of design documentation for one system may be totally inappropriate for meeting another system's needs. The format chosen by the vendor for presenting the design documentation may be influenced by business concerns other than expeditious security evaluation. Different design documentation formats present different advantages and challenges to evaluators and different advantages and costs to vendors that should be weighed.

A system's design may be evolutionary, resulting from improvements that build upon an initial version. Maintaining documentation on a system in a release/update form may be convenient for a developer. However, it is difficult for new developers and life-cycle personnel to gain an understanding of the overall system architecture from documentation that describes the system in chronological terms through system releases and updates. To be useful, these updates shall be incorporated into the design documentation, and the design documentation shall be presented as a complete description of the system, and not the initial description plus supplemental sections describing changes.

3. MEETING THE CRITERIA REQUIREMENTS

This section lists the TCSEC requirements for design documentation at each class. All of these requirements have been extracted from the TCSEC design documentation requirements and include explicit and implicit design documentation requirements, where necessary. Each numbered requirement is referenced in the discussions that follow in Sections 6 and 7 of this document. This section serves as a quick reference for TCSEC class requirements.

As the TCSEC evaluation class level increases, it is implicitly required that the design documentation be more detailed. This is due to an increase in assurance required at the higher classes, as well as the introduction of new features at the higher classes that need to be documented, for example, labeling, auditing.

3.1 The C1 Design Documentation Requirements

Requirement 1 - Describe the philosophy of protection.

Requirement 2 - Describe how the philosophy of protection is translated into the TCB.

Requirement 3 - Describe how the TCB is modularized (if modular).

Requirement 4 - Describe all interfaces between the TCB modules (if modular).

Requirement 5 - Describe how the TCB protects itself.

Requirement 6 - Provide a statement of the system security policy.

3.2 The C2 Design Documentation Requirements

No new requirements have been added at the C2 class.

3.3 The B1 Design Documentation Requirements

Requirement 7 - Provide an informal or a formal description of the security policy model enforced by the TCB.

Requirement 8 - Explain the sufficiency of the security policy model to enforce the security policy.

Requirement 9 - Identify and describe the TCB protection mechanisms.

Requirement 10 - Explain how the TCB mechanisms satisfy the security policy model.

3.4 The B2 Design Documentation Requirements

Requirement 11 - Describe how the TCB is modularized.

Requirement 12 - Describe all of the interfaces between the TCB modules.

Requirement 13 - Provide a formal description of the security policy model.

Requirement 14 - Prove the sufficiency of the security policy model to enforce the security policy.

Requirement 15 - Show that the Descriptive Top Level Specification (DTLS) is an accurate description of the TCB interface.

Requirement 16 - Describe how the TCB implements the Reference Monitor Concept.

Requirement 17 - Describe why the reference monitor is tamper resistant.

Requirement 18 - Describe why the reference monitor cannot be bypassed.

Requirement 19 - Describe why the reference monitor is correctly implemented.

Requirement 20 - Describe how the TCB is structured to facilitate testing.

Requirement 21 - Describe how the TCB is structured to enforce least privilege.

Requirement 22 - Present the results and methodology of the covert channel analysis.

Requirement 23 - Describe the tradeoffs involved in restricting covert channels.

Requirement 24 - Identify all auditable events that may be used in exploitation of known covert storage channels.

Requirement 25 - Provide the bandwidths of known covert storage channels whose use is not detectable by auditing mechanisms.

3.5 The B3 Design Documentation Requirements

Requirement 26 - Identify all auditable events that may be used in exploitation of known covert timing channels.

Requirement 27 - Provide the bandwidths of known covert timing channels whose use is not detectable by auditing mechanisms.

Requirement 28 - Describe how the system complies with additional B3 system architecture requirements, for example, minimal TCB and layering.

Requirement 29 - Informally show consistency of the TCB implementation (in hardware, firmware, and software) with the DTLS.

Requirement 30 - Informally show correspondence between elements of the DTLS and elements of the TCB.

Requirement 31 - Informally show consistency of the DTLS with the model.

3.6 The A1 Design Documentation Requirements

Requirement 32 - Informally show consistency of the TCB implementation with the Formal Top Level Specification (FTLS).

Requirement 33 - Informally show correspondence between elements of the FTLS and elements of the TCB.

Requirement 34 - Clearly describe hardware, software, and firmware internal to the TCB that is not dealt with in the FTLS.

Requirement 35 - Informally or formally show consistency of the FTLS with the model.

Requirement 36 - Informally show correspondence between the FTLS and the DTLS.

4. COMPONENTS OF DESIGN DOCUMENTATION

Design documentation describes why a system is trusted, how this trust is achieved, the mechanisms which provide the trust, and the relevant information that makes proper maintenance of a system possible. Design documentation at TCSEC class C1 lays the foundation for trusted systems by defining the philosophy of protection of a system. As the TCSEC classes increase, the level of detail and the quantity of information contained in the design documentation shall also increase. The following sections discuss design documentation and its role in describing the security policy of the system, the protection mechanisms of the system, and the specific requirements concerning covert channels.

4.1 Documenting The Security Policy

The design and development of any trusted system, from TCSEC class C1 to A1, is based upon a philosophy of protection that shall be described in the design documentation (Requirement 1). Design documentation explains and defines the philosophy of protection by describing how a system provides trust. Trust in computer systems is provided by the protection mechanisms contained within the TCB, such as discretionary access controls and identification and authentication mechanisms. These and all of the TCB mechanisms and their functions shall be described in the design documentation. In addition, the system security policy, i.e., what is being accessed by whom or from what, shall also be described in the design documentation (Requirement 6).

In order to describe how a system is trustworthy, the design documentation shall describe how the philosophy of protection is translated into the TCB (Requirement 2) and how it is supported by the TCB protection mechanisms. The design documentation shall first define the boundaries of the system and shall describe the parts of the system that are security relevant and the parts that are not. Rationale shall be presented that those portions of the system which are claimed to be outside of the TCB, are really outside. The proper identification of these parts is important to the maintenance of security in the system because it is necessary to know when a change to the system will affect the TCB implementation, and possibly violate the security policy of the system.

At the higher TCSEC classes, the description of the philosophy of protection evolves into a more structured description of how a system provides trust. At TCSEC class B1, this philosophy of protection shall be presented as an informal or formal security policy model in the design documentation (Requirement 7). This security policy model shall informally or formally define the subjects, objects, modes of access, and the security properties of the system. In addition, the model shall define the initial state of the system, a secure state of the system, and the way in which the system progresses from one state to the next. An informal security policy model may be presented in a natural language, for example, English. An explanation shall be provided demonstrating that the informal model is sufficient to enforce the security policy (Requirement 8).

At TCSEC class B2, a formal security policy model shall exist (Requirement 13). In addition to the B1 requirements, the formal security policy model shall contain: a set of security properties that captures the security policy, an abstract description of the operations performed by the TCB, and a rigorous argument through the use of predicate calculus that the description is consistent - internally consistent, that is, is not self-contradictory. The model shall include a proof that if the initial state of the system satisfies the definition of a "secure" state and if all assumptions of the model are satisfied, then all future states of the system will be secure.

A security policy model provides assurance that the system has been designed to enforce the security policy and provides a basis for the TCB implementation. As a means of increasing assurance, the design documentation shall show that the security policy model is sufficient to enforce the security policy of the system (Requirement 8). At TCSEC class B1, it shall be sufficient to show this in a natural language, e.g., English, but at class B2, this sufficiency of the security policy model shall be shown through a formal proof (Requirement 14). The design documentation shall provide a mapping of the security properties to the security policy. This sufficiency shall be demonstrated by describing how all aspects of the security policy are addressed by the security policy model.

An example of a formal security policy that enforces the DoD security policy is the Bell-La Padula model [1]. Although the Bell-La Padula security policy model supports the DoD security policy, it is important to realize that this does not mean that the Bell-La Padula model model can be directly used for all systems. The Bell-La Padula model, when used with different systems, will need to be representative of the system.

At TCSEC class B2, the TCSEC design specification and verification requirement calls for a descriptive top level specification (DTLS) of the TCB to be maintained to provide documentary evidence of how the formal security policy model is implemented through the TCB interface. The DTLS provides evaluators with a better understanding of the implementation of the reference monitor and provides maintenance personnel with the necessary documentation to correct, modify, or augment the TCB without destroying the TCB's cohesiveness and internal consistency. The description of the TCB should contain a description of the services and functions provided by the TCB and how they are invoked. For example, for UNIX1-based systems, the DTLS may be based upon enhanced manual pages. The manual pages shall include enough information to satisfy the design specification and verification requirement that the TCB be described in "terms of exceptions, error messages, and effects."[5] These individual manual sections should be accompanied by detailed section headers which clearly explain the security concepts and entities referenced within each section. The design documentation shall demonstrate that the DTLS is an accurate description of the TCB interface (Requirement 15). It should do this by accurately and completely describing the DTLS in relation to the TCB interface.

The design documentation shall be used to test against the TCB to, "demonstrate that the TCB implementation is consistent with the descriptive top level specification." [5] The design documentation shall describe how the TCB is structured to facilitate this testing (Requirement 20).

At TCSEC class B3, the design documentation shall informally show consistency of the TCB hardware, firmware, and software implementation with the DTLS as well as showing correspondence between elements of the DTLS and elements of the TCB (Requirements 29, 30). The goal of these two requirements is to ensure that the mapping between the TCB and DTLS is complete, easily understandable, unambiguous, and one-to-one between elements of the TCB implementation and elements of the DTLS. The level of detail of this mapping should be sufficient for any inconsistency to be obvious to members of the vendor's development team throughout the system life-cycle.

Also at TCSEC class B3, the design documentation shall provide a mapping from the DTLS to the TCB implementation (Requirement 30). This mapping should demonstrate that all elements that were specified were, in fact, implemented, and that any code that appears which is not specified directly merely reflects implementation detail; that there are no new, unspecified, user interfaces implemented. With this mapping, the design documentation shall describe all areas of correspondence between elements of the -

----

1 UNIX is a trademark of AT&T Bell Labs

DTLS and elements of the TCB. For example, the mapping may be pointers in the DTLS description to source code modules of the TCB implementation.

The addition of the design specification and verification requirement of a mapping between the DTLS and the formal security policy model completes the evidence from the security policy to implementation. The entities of the model shall be shown to correspond to the elements of the DTLS (Requirement 31). This correspondence provides assurance that the security properties that are proven in the formal model are accurately reflected in the implementation.

At TCSEC class A1, the mapping shall be from the formal top level specification (FTLS) to the TCB (Requirements 32, 33). In addition, the mapping to the model shall be from the FTLS (Requirement 35). These changes reflect the introduction of an FTLS requirement in the design specification and verification requirements at A1. The design documentation shall describe how the FTLS accurately represents the TCB interface. The hardware/firmware components of the TCB, such as mapping registers and direct memory access input/output (I/O) components, that are directly or indirectly visible at the TCB interface shall be described in the design documentation. As stated previously, the goal of these requirements is to ensure that the mapping between the elements of the TCB implementation and the FTLS are complete, easily understandable, unambiguous, and one-to-one.

Although the TCSEC design documentation requirement changes at class A1 to require a mapping from the FTLS to the TCB, a DTLS is still required for class A1 systems. At TCSEC class A1, the DTLS serves to augment the FTLS by completing the description of the TCB in an informal language and by providing the conceptual glue to the specification of the reference monitor mechanism and the other TCB components. Since there is an explicit requirement at class A1 that both the FTLS and DTLS correspond to the formal security policy model, the DTLS and the FTLS must correspond (Requirement 36). At TCSEC class A1, "the FTLS and DTLS may be two separate documents, or they may be combined into a Complete Top Level Specification (CTLS). In a CTLS, the FTLS and DTLS portions (shall) be separately identifiable. The CTLS (shall) be a complete and accurate description of the TCB, and it (shall) be sufficiently well commented/annotated so that it can be easily understood with little or no knowledge of formal specifications."[8]

It is recognized that not all of the TCB internals are able to be specified within the FTLS. For the hardware, firmware, and software internal to the TCB, but not dealt with in the FTLS, the design documentation shall describe them in complete, clear, and careful detail (Requirement 34).

4.2 Documenting TCB Protection Mechanisms

As part of the description of the philosophy of protection and how it translates into the TCB, the design documentation shall include explanations of the security services offered by the TCB software, hardware, and firmware mechanisms from a system level view (Requirement 2). At TCSEC class C1, the design documentation for these protection mechanisms shall include how the mechanisms protect the TCB from tampering (Requirements 5). The description of why the TCB is tamper resistant is an important requirement for all of the TCSEC classes. This design documentation requirement supports the TCSEC class C1 system architecture requirement which calls for the TCB to maintain a domain that implements the reference monitor concept that, "protects it from external interference or tampering"[5]. The mechanisms described in this section of the design documentation include things such as Discretionary Access Control (DAC) and identification and authentication (I&A) mechanisms. For example, the design documentation shall describe the DAC enforcement mechanism and how it controls discretionary access between named users or groups and named objects within the ADP system. As it relates to identification and authentication, the design documentation shall describe how users are identified to the TCB and the mechanism that authenticates the user's identity. Furthermore, the design documentation shall describe how the TCB protects the authentication data. To ensure that these mechanisms have not failed in any way, hardware and software mechanisms shall exist to periodically validate the correct operation of the on-site hardware and firmware elements of the TCB. These system integrity mechanisms shall also be described in the design documentation.

At TCSEC class B1, the design documentation shall identify and provide descriptions of the TCB protection mechanisms (Requirement 9). This documentation is required to provide the additional assurance required at TCSEC class B1. In most cases, these TCB protection mechanisms at TCSEC class B1 may be the same protection mechanisms that were described in TCSEC class C1, but at class B1, the description of these mechanisms shall describe how they support the additional system architecture requirement for process isolation. Process isolation mechanisms that prevent untrusted subjects from directly accessing separate address spaces are introduced at TCSEC class B1 and shall be described in the design documentation. The design documentation shall also show that all of the security services required by the security policy model are provided by the TCB mechanisms (Requirement 10).

At TCSEC class B2, the design documentation shall describe how the TCB protection mechanisms implement the reference monitor concept, i.e., is nonbypassable, always invoked, and small enough to be analyzed (Requirement 16). The design documentation requirement should demonstrate how the reference validation mechanism is tamper resistant, cannot be bypassed, is correctly implemented, and is structured to enforce least privilege (Requirements 17, 18, 19, 21). Although a reference monitor has been in place since TCSEC class C1, the system architecture requirements at TCSEC class B2 require that the TCB be protected, "from external interference and tampering," maintain process isolation, and be "internally structured into well defined largely independent modules."[5] These additional requirements shall be reflected in the design documentation.

One position for how the reference monitor hardware should be documented is presented in the following paragraphs:

"For microprograms (firmware), design documentation is needed for common routines, that is, documentation which fully describes the functionality and what is done to implement that functionality. At the least, a high level view of major operations, e.g., interrupts, I/O instruction interpretations is needed if the microcode is not modular enough to be described in terms of microroutines.

For items inside the TCB, but outside of the reference monitor (such as most disk controllers, printers, and other peripherals), the interface must be described, but not the internals.

In the case of systems that do not use microcode, convincing arguments must be provided as to what elements of the hardware are security critical and why."[2]

The design documentation for the TCB firmware should parallel the documentation that is written for the TCB software, that is, it should fully describe the functionality and what is done to implement the functionality of the security kernel.

Assurance needs to be provided that the TCB is protected from modification. At TCSEC class B2, the design documentation shall provide this assurance through a description of why the reference monitor is tamper resistant (Requirement 17). This description shall include the methods and mechanisms by which the TCB protects itself from illicit modification. Any hardware mechanisms used by the TCB to separate protection critical elements from those that are not protection critical shall be described. The mechanisms used by the TCB to help support logically distinct storage objects with separate attributes shall also be described. The mechanisms used to protect against illicit modification may include some of the same mechanisms used to mediate accesses of objects by subjects that were introduced at TCSEC class C1. These mechanisms shall be described again at TCSEC class B2, but in greater detail as to how they apply to the reference validation mechanism.

The previous paragraph explained how the design documentation describes protection mechanisms, but more importantly, at TCSEC class B2, the design documentation shall show that all of the TCB software, firmware, and hardware mechanisms have been implemented as described and that the implementation functions correctly (Requirement 19). The design documentation shall justify the correctness of the entire TCB.

Also, at TCSEC class B2, the design documentation shall describe how the TCB is structured to enforce least privilege (Requirement 21). This description shall relate to the hardware, firmware, and software modules of the TCB, as well as to the enforcement of least privilege both within the TCB and upon trusted subjects. Least privilege ensures that any TCB module or trusted process has only those privileges and capabilities needed for it to perform the specific function for which it was designed. For example, if the hardware architecture implements protection rings, a description shall be given of the ring mechanisms. This description shall show how access to the innermost ring provides a means of running highly privileged processes, while the outermost ring provides a means of running unprivileged processes. Likewise, the description shall justify placement of functions within the higher privileged rings and the conferring of special privileges to trusted processes. Thus, the hardware is shown as a means of enforcing least privilege.

Similarly, firmware and software mechanisms may provide a means of enforcing least privilege. For example, a labeling mechanism may be implemented in software or firmware. Because labels may be used to enforce least privilege, the software or firmware modules enforcing the labeling and label based access control shall be shown as a means of enforcing least privilege.

The separation of administrative roles in the system is one more way in which least privilege may be exercised. In this case, the roles of system administrator, security administrator, and/or system auditor may be performed by separate individuals. This is to ensure that the security functions of the system are not able to be performed by a single person. The way these roles are carried out in the system shall be described in the design documentation.

At TCSEC class B3, the system architecture requirements call for the TCB to be minimized, i.e., only security relevant functions appear within the TCB. The TCB at this class, "shall incorporate significant use of layering, abstraction, and data hiding," and shall have minimal complexity. The design documentation shall describe how the system complies with these additional architectural requirements in (Requirement 28). As stated previously, as the TCSEC classes increase and the implementation of the reference monitor concept becomes more defined, the amount of design documentation shall also increase.

4.3 Documentation of Covert Channels

A portion of the B2 requirements for design documentation addresses covert channels. The results of all covert channel analysis need to be in the design documentation to aid in the design and development of TCB mechanisms. For this reason, the design documentation shall present the results of the covert channel analysis and the methodology used (Requirement 22). The design documentation shall provide an overview of the covert storage channel analysis and testing procedures. It shall document the results of these tests and all of the covert channels identified. All auditable events shall be identified and described for all covert storage channels that are not removed from the system (Requirement 24).

When covert channels are identified, actions are sometimes taken to restrict the bandwidth of those channels. The design documentation shall describe and discuss these actions and the resulting degree of covert channel restriction in light of performance degradation, operational utility, or other considerations (Requirement 23). Processing delays resulting from reducing the number and bandwidth of covert channels shall be identified and characterized. The design documentation shall also note whether the exploitation of known covert channels is auditable. There will be some covert storage channels whose use will not be detectable by auditing mechanisms. The design documentation shall document the worst case and expected case bandwidths of these storage channels whose exploitation is not auditable (Requirement 25).

At TCSEC class B3, the design documentation shall recognize the introduction of covert timing channels into the requirements and shall consider them in all covert channel related descriptions as stated above (Requirements 26, 27). The covert timing channel analysis and testing procedures, and the results obtained from the tests shall be described in the design documentation. Additionally at TCSEC class A1, formal methods shall be used in the covert channel analysis and shall be described in the design documentation.

5. OTHER TOPICS

5.1 Modularity

An important architectural feature of trusted systems for TCSEC class B2 and above is that the TCB be modular. The modularity of the TCB is important for ease of understanding, ease of analysis, and ease of maintenance. Modularity ensures that interfaces are well defined and errors are contained. It also provides a basis for enforcing least privilege. The content of hardware and software modules should be selected based on the following criteria: a module performs exactly one well defined action, a module has a well defined interface, a module interacts with other modules only in well defined ways, and a module is called upon to perform a function whenever that function is required. Although TCB modularity is not a requirement until class B2 (Requirements 11, 12), it is possible that vendors would want to build systems with modular TCBs at the lower classes. Regardless of class, if the TCB is modular, the design documentation shall describe how the TCB is modular and the interfaces between the TCB modules (Requirement 3, 4). As with all design documentation, the level of detail shall permit the description of the interfaces between the modules to be a useful description. Specifically, the design documentation shall include identification of the TCB hardware, software, and firmware modules, why the modules are considered as such, the interfaces between them, and the implementation of the modules. A mapping of the security services and mechanisms to the modules should also be described.

The level of detail of the design documentation increases the amount of assurance to be gained by the developers and evaluators. The description of the interfaces between the modules, whether hardware, firmware, or software, shall describe the types and sources of information passing between them (Requirement 4). In addition, the interfaces between these TCB modules and other system modules external to the TCB shall be described. This description is necessary to show that no breach of security can occur through the interfaces.

In some cases, software modules may depend upon hardware or firmware modules to perform correctly and these dependencies should also be included in the design documentation.

5.2 Hardware Design Documentation

Hardware design documentation for a system shall be provided at all levels of trust. Specifically, at TCSEC classes B2 and above, it is felt that the hardware design documentation is critical to the security of a system. At this class, systems "shall make effective use of available hardware to separate those elements that are protection critical from those that are not."[5] To meet this TCSEC requirement, developers should know the hardware base they are building on top of. Also, evaluators will need the hardware design documentation in order to evaluate that the vendor is making "effective use" of the hardware.

The hardware design documentation includes descriptive information about the system's Central Processing Unit(s) (CPU), Memory Management Unit(s) (MMU), and all other additional processors, for example, I/O Processors, channel devices. The hardware design documentation is intended to discuss what the hardware is meant to do, but does not need to include details of implementation, such as the flow of control to perform a specific action. The hardware design documentation for every logical module in the hardware base should include a functional name, functional description, and a functional interface of that module.

The hardware design documentation defines the hardware portion of the TCB interface. The information on the hardware interface is important to correctly develop TCB routines and device drivers. Additionally, the hardware design documentation provides sufficient information that the TCB meets the System Architecture requirements of the TCSEC.

The information contained in the hardware design documentation shall be complete, specifying all possible interfaces to the system hardware, including the user-to-hardware interface as well as the TCB software-to-hardware interface. The hardware design documentation should include unprivileged instructions, privileged instructions, unpublished instructions, and all CPU-to-MMU, CPU-to-

Channel, CPU-to-I/O bus, and additional processor interactions. Also, the software interface visible registers that exist on the CPU, MMU, and other processors shall be described in the hardware design documentation.

The design documentation for some hardware modules may require internal detail down to a bit level functional description of the module. The modules that fall into this category are those that make up thereference monitor, such as the address translation module, process isolation support module, fault handling module, I/O control module, and the diagnostic module. These hardware modules of the reference monitor directly support security and will require an explanation of why the reference monitor is "tamper resistant, cannot be bypassed, and is correctly implemented."[5]

5.3 Configuration Management

The design documentation for the system shall be under configuration management for the entire life-cycle of the system. Design documentation is only useful if it is complete and accurate. This means that any change to the system should also result in a change to the design documentation for the system.

The design documentation for a system should be treated as a configuration item for the system and should be subject to the identification, control, accounting, and audit functions of configuration management. "Initial phases of configuration control are directed towards control of the system configuration as defined primarily in design documents.

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 nonexistent."[7]

The TCSEC requirements for configuration management do not begin until TCSEC class B2, but this should not mean that the design documentation for TCSEC class C1 to B1 systems not be under some type of control. At these lower classes, the control process for the design documentation may be less formal than that required by the configuration management requirements, but it should still provide assurance that the design documentation accurately describes the current system.

The National Computer Security Center has recently developed the Ratings Maintenance Program (RAMP) which requires configuration management at these lower classes of trust. "By training vendor personnel to recognize which changes may adversely affect the implementation 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 reevaluate the new version." [7]

For further information about the RAMP program and about the configuration management requirements for RAMP, contact:

National Computer Security Center

9800 Savage Road

Fort George G. Meade, MD 20755 6000

Attention: C12

6. SUMMARY OF DESIGN DOCUMENTATION

Design documentation is responsible for describing systems at all levels of trust. During the life-cycle of a system, it describes the system to facilitate changes and maintenance of the system. As it relates to the security of a system, design documentation provides assurance by describing how a system provides trust and shows that all of the protection mechanisms of a system are correctly implemented and sufficiently provide the needed trust. At the lower classes, design documentation begins to describe how security is provided in a system by stating the philosophy of protection of the system. At TCSEC class B1, the design documentation describes the security policy model of a system, and at TCSEC class B2 the security policy model is required to be formal.

Many of the other requirements in the TCSEC are related to design documentation. Design documentation shall describe how these requirements are satisfied. Covert channels are specifically addressed in the design documentation requirement. The assurance provided by design documentation is dependent upon its thoroughness and accuracy. When design documentation is written, the role that it plays in the system life-cycle should be kept in mind. A new employee should be able to look at the design documentation and get an understanding of what the current system is and how it works. The key word in design documentation is current. When a system changes, the design documentation shall change accordingly. By accurately describing a system, design documentation provides assurance that there is an understanding of how and why the system provides trust. In addition, it provides information that will enable developers to analyze changes to the system to ensure that they do not adversely affect the trustworthiness of the system.

APPENDIX B

EXCERPTS FROM FINAL EVALUATION REPORTS

This appendix reproduces excerpts from Final Evaluation Reports for products currently on the Evaluated Products List. The excerpts are reproduced from the "Applicable Features" portion of the section describing how the product met the requirements for Design Documentation.

The Final Evaluation Reports are available from the National Computer Security Center. However, most of the vendor documents mentioned in this appendix contain proprietary information, and therefore are not publicly available. Please do not request copies of the vendor documents from the National Computer Security Center.

B.1 CLASS C2

B.1.1 UTX/32S[6]

The following documents were provided to the evaluation team in fulfillment of the Design Documentation criterion:

"Security Policy Model"

"Program Maintenance Manual UTX/32S, Release 1.0 (DRAFT)".

"System Calls" and "Maintenance" in the "System Administrator's Reference Manual".

"4.2BSD and UTX-32 Differences Study for Gould

UTX/32S"

"Memory Management for Gould UTX/32S"

"Object Reuse Study for Gould UTX/32S"

The "Gould UTX/32S 1.0 Security Policy Model" describes Gould's philosophy of protection and explains how this philosophy is translated into the TCB. It identifies all elements comprising the TCB, including the kernel, programs, data files, and processes. Subjects and objects are identified, and the mediation of accesses between them is described. A mapping from the TCB to the security philosophy is provided, and the discretionary access control, identification and authentication, and audit features and mechanisms are described. Additionally, the document discusses the role of secure sockets in interprocess communications. The "Gould UTX/32S 1.0 Security Policy Model" identifies all programs comprising the TCB.

The kernel interface is described by the "System Calls" section of the "System Administrator's Reference Manual". The "Maintenance" section of the reference manual comprises manual pages useful for systems programmers in maintaining UTX/32S. "4.2BSD and UTX-32 Differences Study for Gould UTX/32S" describes differences between 4.2BSD UNIX and Gould UTX/32

1.2. Using "4.2BSD and 4.3BSD as Examples of the UNIX System," by J.S. Quarterman, A. Silberschatz, and J.L. Peterson (Computing Surveys, Vol. 17, No. 4, December 1985, pp. 379-418), as a baseline, the document identifies all instances where Gould UTX/32 differs from the described UNIX system.

The "UTX/32S Program Maintenance Manual" describes code modifications made to UTX/32 to meet the requirements of the "Gould UTX/32S 1.0 Security Policy Model". The document includes an overview of the mechanisms implemented in UTX/32S to strengthen security and to correct problems found in UTX/32 and other UNIX systems, and detailed descriptions for: the implementation of trusted servers to replace the functionality of the eliminated setuid and setgid bits; kernel modifications; auditing mechanisms; and additions, deletions, and modifications to utilities and libraries. Each module description includes an overview, a functional specification, and a design specification. Pointers to source code, which Gould made available to the evaluation team are provided.

Security critical features of the Gould PowerNode hardware used by UTX/32S are described in "UTX/32S Traps and Interrupts and Memory Management for Gould UTX/32S." "UTX/32S Traps and Interrupts" describes how UTX/32S makes use of the trap and interrupt facilities to interface with the hardware and process environments. "Memory Management for Gould UTX/32S" describes how UTX/32S uses the memory management facilities of the PowerNode hardware to provide the process environment. Both documents include applicable material from "Gould SS 6 (Virtual Mode), V6, and V9 Central Processing Unit Reference Manual".

"Object Reuse Study for Gould UTX/32S" provides details regarding how UTX/32S hardware and software manage system objects. This study identifies the system resources which can be allocated and deallocated, and details the strategies used to ensure that one process cannot gain access to the resources or data previously allocated to another process. This study, along with "Memory Management for Gould UTX/32S", provides a good description of UTX/32S design features which are used to meet the Object Reuse criterion.

B.2 CLASS B2

B.2.1 Multics3]

The following documents satisfy the Design Documentation requirement:

Applicable Features

The "Computer Security Model: Unified Exposition and MULTICS Interpretation" provides a description of Honeywell's philosophy for protection and how this is translated into the TCB. The security model enforced by the TCB is the Bell-La Padula model.

Multics has a set of Multics Design Documents (MDDs) that describe the TCB. (These documents are Honeywell Internal Documentation and are available only through the vendor by request. Honeywell reserves the right to deny such requests.) The MDDs are organized by major TCB service or function. These design documents describe the interfaces between TCB modules, how the TCB implements the reference monitor, and how the TCB is structured to facilitate testing and enforce least privilege.

These documents coupled with the Honeywell produced "Multics Interpretation," referenced in the previous paragraph identify the security protection mechanisms and explain how they satisfy the model.

The DTLS is an accurate description of the TCB interface.

The "Covert Channel Analysis" describes all identified covert channels, how they can and cannot be restricted, how they are audited, and their bandwidths.

B.3 CLASS A1

B.3.1 SCOMP [4]

The following documents satisfy the Design Documentation requirement: The manufacturer's philosophy of protection is documented in "SCOMP: A Solution to the Multilevel Security Problem" and its translation into the TCB given in "SCOMP Trusted Computing Base." The interfaces between the TCB modules are described in the several Part II specifications,

"Detail Specification for SCOMP Kernel Part I,

Release 2.1"

"Detail Specification for SCOMP Kernel Part II,

Release 2.1"

A formal description of the security policy model (Bell-La Padula) that is enforced by the TCB is given in "Secure Computer Systems", for the general case and Multics in particular in "Computer Security Model:

Unified Exposition and MULTICS Interpretation". The Bell-La Padula Model has been accepted by the National Computer Security Center to model security policy "Security Requirements for Automatic Data Processing Systems" and to be consistent with its axioms. An interpretation of the model for the SCOMP system is given in "SCOMP Interpretation of the Bell-La Padula Model."

The specific TCB protection mechanisms are 1) protection rings, 2) SPM mediation of per user virtual memory, 3) minimum privilege for each TCB function, 4) integrity levels for users, operators, administrators, and security administrators, 5) individual trusted software processes for separate functions, and 6) ring gates and checks on parameter passing. The Part II Specifications previously referenced provide the necessary documentation for satisfaction of this requirement. The explanation given to show that the TCB protection mechanisms satisfy the model appears in "SCOMP Interpretation of the Bell-La Padula Model."

Section 3 of "SCOMP Trusted Computing Base" describes the SCOMP TCB reference monitor implementation. An analysis of the Reference Monitor appears in Appendix C and concludes that the informal proofs that the SCOMP system implements the reference monitor concept are adequate.

The TCB implementation was shown to be consistent with the FTLS by specification to source code mappings

"FTLS to Code Mapping for the SCOMP Kernel Software"

"FTLS to Code Mapping for SCOMP Trusted Software"

"Justification for Unspecified Code for the SCOMP Kernel Software, Release 2.1"

"Justification for Unspecified Code for SCOMP Trusted Software"

TCB testing is documented in:

"SCOMP Kernel Test Procedures"

"SCOMP Kernel Functional Test Summary"

"Kernel Software Test Report for the SCOMP,

Release 2.1"

"Trusted Software Test Plan for the SCOMP"

"Trusted Software Test Report for the SCOMP,

STOP Release 2.0"

"Trusted Software Test Report for the SCOMP,

Appendix A:Test Programs, Appendix B: Test Results"

"SCOMP Test and Verification Software

Description, Rev. 3"

The TCB structure provided added assurance of the validity of the testing and helped to demonstrate the implementation of least privilege. The results of the covert channel analysis including conservative bandwidth estimates are presented in "Covert Channels in the SCOMP Kernel" and "Flow and Covert Channel Analysis for SCOMP Trusted Software, Release 2.1." 61 Auditable events, identified in Section 13 of "SCOMP Trusted Facility Manual, STOP Release 2.1", and the scheme of randomly selected delays on exception returns appear to satisfactorily limit the utility of the identified covert channels.

Finally, the internal TCB mechanisms that are not security related and hence not dealt with in the FTLS are described in the commercial Honeywell Level 6 documentation ("Honeywell Level 6 Minicomputer Systems Handbook, CC71" and "Honeywell Level 6 Communications Handbook, AT97-02D"), and the SCOMP system unique specifications:

"Detail Specification for SCOMP Kernel Part I,Release 2.1"

"Detail Specification for SCOMP Kernel Part II, Release 2.1".

GLOSSARY

Access

A specific type of interaction between a subject and an object that results in the flow of information from one to the other.[9]

Access Attribute

Characteristic of an access of an object that specifies possible results of the access. Four example access attributes follow: execute (processing based upon the object accessed, but neither altering nor viewing capability); read (viewing but not altering capability); append (altering but not viewing capability); and write (both altering and viewing capabilities).[1]

Audit Trail

A chronological record of system activities that is sufficient to enable the reconstruction, reviewing, and examination of the sequence of environments and activities surrounding or leading to an operation, a procedure, or an event transaction from its inception to final results.[9]

Covert Channel

A communication channel that allows two cooperating processes to transfer information in a manner that violates the system's security policy. Also called confinement channel.[9]

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.[5]

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.[5]

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.[5]

Formal Security Policy Model

A mathematically precise statement of a security policy. To be adequately precise, such a model must represent the initial state of a system, the way in which the system progresses from one state to another, and a definition of a "secure" state of the system. To be acceptable as a basis for a TCB, the model must be supported by a formal proof that if the initial state of the system satisfies the definition of a "secure" state and if all assumptions required by the model hold, then all future states of the system will be secure. Some formal modeling techniques include: state transition models, temporal logic models, denotational semantics models, algebraic specification models. An example is the model described by Bell-La Padula.[9]

Formal Top Level Specification (FTLS)

A top level specification that is written in a formal mathematical language to allow theorems showing the correspondence of the system specification to its formal requirements to be hypothesized and formally proven.[9]

Least Privilege

This principle requires that each subject in a system be granted the most restrictive set of privileges needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use.[9]

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, and network nodes.[9]

Reference Monitor Concept

An access control concept that refers to an abstract machine that mediates all accesses to objects by subjects.[9]

Security Kernel

The hardware, firmware, and software elements of the Trusted Computing Base that implement the reference monitor concept. It must mediate all accesses, be protected from modification, and be verifiable as correct.[9]

Security Level

The combination of hierarchical classification and a set of nonhierarchical categories that represents the sensitivity of information.[9]

Security Mechanism

A system or means of implementing a security service within a system.

Security Policy

The set of laws, rules, and practices that regulate how an organization manages, protects, and distributes sensitive information.[9]

Security Policy Model

A formal presentation of the security policy enforced by the system. It must identify the set of rules and practices that regulate how a system manages, protects, and distributes sensitive information.[9]

Security Service

A system or method of providing a security relevant feature in the system.

Sensitivity Label

A piece of information that represents the security level of an object. Sensitivity labels are used by the TCB as the basis for mandatory access control decisions.[9]

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.[9]

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. It creates a basic protection environment and provides additional user services required for a trusted computer system. The ability of a trusted computing base to correctly enforce a security policy depends solely on the mechanisms within the Trusted Computing Base and on the correct input by system administrative personnel of parameters (e.g., a user's clearance level) related to the security policy.[9]

REFERENCES

1. Bell, D.E., and L.J. La Padula, "Secure Computer System: Unified Exposition and Multics Interpretation," MTR-2997, Mitre Corp., Bedford, MA, July 1975.
2. Gabriele, Mark D., Excerpts from the Criteria Discussions Forum on the Dockmaster computer system at the National Computer Security Center, Forum entry #0680, Hardware Design Documentation at B2 and Above, May 9, 1987.
3. National Computer Security Center, Final Evaluation Report of Honeywell Multics MR11.0, CSC-EPL-85/003, June 1, 1986.
4. Department of Defense Computer Security Center, Final Evaluation of

SCOMP, Secure Communications Processor, STOP Release 2.1, CSC-EPL-85/001, September 23, 1985.

5. Department of Defense Standard, Department of Defense Trusted Computer System Evaluation Criteria, DoD 5200.28- STD, December 1985.

6. National Computer Security Center, Final Evaluation Report of Gould, Inc., Computer Systems Division, UTX/32S, Release 1.0, CSC-EPL-86/007, December 31, 1986.

7. National Computer Security Center, A Guide to Understanding Configuration Management in Trusted Systems, NCSC-TG-006, March 28, 1988.

8. National Computer Security Center, Criterion Interpretation, Report No. C1-CI-01-87, 1987.

9. National Computer Security Center, Glossary of Computer Security Terms, NCSC-TG-004-88, October 1988.