A GUIDE to UNDERSTANDING OBJECT REUSE in TRUSTED SYSTEMS

Patrick R. Gallagher, Jr. July 1992
Director
National Computer Security Center

Table of Contents

NCSC-TG-018
Library No. 5-223,170
Version 1

FOREWORD

A Guide to Understanding Object Reuse in Trusted Systems provides a set of good practices related to object reuse. We have written this guideline to help the vendor and evaluator communities understand the requirements for object reuse as described in the Department of Defense Trusted Computer System Evaluation Criteria. In an effort to provide guidance, we make recommendations in this technical guideline that are not requirements in the Criteria.

The Object Reuse Guide is the latest in a series of technical guidelines published by the National Computer Security Center. These publications provide insight to the Trusted Computer System Evaluation Criteria requirements for the computer security vendor and technical evaluator. The goal of the Technical Guideline Program is to discuss each feature of the Criteria in detail and to provide the proper interpretations with specific guidance.

The National Computer Security Center has established an aggressive program to study and implement computer security technology. Our goal is to encourage the widespread availability of trusted computer products for use by any organization desiring better protection of its important data. One way we do this is by the Trusted Product Evaluation Program. This program focuses on the security features of commercially produced and supported computer systems. We evaluate the protection capabilities against the established criteria presented in the Trusted Computer System Evaluation Criteria. This program, and an open and cooperative business relationship with the computer and telecommunications industries, will result in the fulfillment of our country's information systems security requirements. We resolve to meet the challenge of identifying trusted computer products suitable for use in processing information that requires protection.

I invite your suggestions for revising this technical guideline. We will review this document as the need arises.

ACKNOWLEDGMENTS

The National Computer Security Center extends special recognition and acknowledgment to James Anderson, COL Rayford Vaughn (USA), and Rick Siebenaler as authors of this document. Patricia R. Toth is recognized for the development of this guideline, and Capt. James A. Muysenberg (USAF) is recognized for its editing and publication.

We wish to thank the many members of the computer security community who enthusiastically gave their time and technical expertise in reviewing this guideline and providing valuable comments and suggestions.

1 INTRODUCTION

1.1 BACKGROUND

The principal goal of the National Computer Security Center (NCSC) is to encourage the widespread availability of trusted computer systems. In support of this goal the NCSC created a metric, the DoD Trusted Computer System Evaluation Criteria (TCSEC) [2], against which computer systems could be evaluated.

The TCSEC was originally published on 15 August 1983 as CSC-STD-001-83. In December 1985 the Department of Defense adopted it, with a few changes, as a Department of Defense Standard, DoD 5200.28-STD. DoD Directive 5200.28, Security Requirements for Automated Information Systems (A 155) [4], requires the TCSEC be used throughout the Department of Defense. The TCSEC is the standard used for evaluating the effectiveness of security controls built into DoD ASs.

The TCSEC is divided into four divisions: D, C, 19, and A. These divisions are ordered in a hierarchical manner, with the highest division (A) being reserved for systems providing the best available level of assurance and security. Within divisions C and 19 are subdivisions known as classes, which are also ordered in a hierarchical manner to represent different levels of security in these divisions.

1.2 PURPOSE

This document is written to help vendors and evaluators understand the object reuse requirement. It also provides guidance to vendors on how to design and incorporate effective object reuse mechanisms into their systems. Some examples for accomplishing object reuse are provided in this document, but they are not the only way to meet the requirement. Nor are the recommendations supplementary requirements to the TCSEC. The only measure of TCSEC compliance is the TCSEC itself.

1.3 SCOPE

This guideline discusses the object reuse requirement applicable to the trusted computing bases (TClBs) found in classes C2 through Al. Although it is directed toward object reuse, this guideline also addresses other TCSEC requirements that tend to strengthen the assurance for the correct performance of object reuse.

1.4 CONTROL OBJECTIVE

The general control objective for object reuse is specified in the TCSEC. It was originally obtained from DoD Directive 5200.28 (April `78 version) which has now been modified and reissued (March `88) but still embodies the spirit of the original requirement. It requires automated information systems (AISs) processing classified information to, "with reasonable dependability, prevent:

a. Deliberate or inadvertent access to classified material by unauthorized persons . . . ."[2, page 59]

The word "unauthorized" in the above objective is deserving of further explanation. Certainly one must assume that strong Identification and Authentication (I & A) mechanisms ensure that users are authorized to be on the system prior to the need for object reuse controls. However, the object reuse mechanisms are designed to ensure that the authorized user of the system does not obtain residual information from system resources.

In this guideline, "classified" means any information requiring protection or authorization to access. The disclosure arising from the absence of an object reuse mechanism is considered inadvertent (assuming all other TCSEC requirements are in place and working properly).

2 OVERVIEW OF PRINCIPLES

Object reuse is defined by A Guide to Understanding Data Remanence in Automated Information Systems as:

> "The reassignment to some subject of a storage medium (e.g., page frame, disk sector, magnetic tape) that contained one or more objects. To be securely reassigned, no residual data can be available to the new subject through standard system mechanisms." [1]

The object reuse requirement of the TC$EC is intended to assure that system resources, in particular storage media, are allocated and reassigned among system users in a manner which prevents the disclosure of sensitive information.

The requirement ultimately derives from observations made during the early penetration testing experiences (circa 1969-1973) that systems typically did not clear a user s working memory (often called "scratch space") when the user completed a session. Thus, when memory was reassigned to another user, the information placed in it by the previous user was now available to the new owner of the scratch space. Often, the disclosure was inadvertent: the second user had not intended to obtain the information suddenly made available. However, it also meant that a malicious (or just curious) user could browse (or "scavenge") through the system, obtaining scratch space and looking for items of interest (the equivalent of rummaging through the trash). Depending upon the details of the system architecture and operating system, a browser could more or less control whose work space was being accessed. Although the problem was first noted as one involving primary storage (or main memory), it clearly extends to secondary storage as well; removable media (such as tapes and removable disks) and disk space must be allocated such as to prevent one user s information from inadvertently becoming available to others as system storage resources are shared among system users.

In short, with effective object reuse mechanisms in place, a penetrator is prevented from obtaining information through browsing attacks (e.g., logging onto the system as a user and browsing). Object reuse does not necessarily protect against physical attacks on the storage medium, especially using sophisticated equipment and methods (e.g., taking a degaussed disk pack and trying to read residual information from the platters). Protection against physical attacks are addressed by A Guide to Understanding Data Remanence in Automated Information Systems [1].

2.1 THE TCSEC OBJECT REUSE REQUIREMENT

The object reuse requirement first appears at class C2 (controlled access protection) and remains unchanged through class Al (verified protection). Although the requirement remains unchanged, the assurance of proper implementation increases as the class increases.

> The requirement is:
> "All authorizations to the information contained within a storage object shall be revoked prior to initial assignment, allocation or reallocation to a subject from the TCB's pool of unused storage objects. No information, including encrypted representations of information, produced by a prior subject's actions is to be available to any subject that obtains access to an object that has been released back to the system." [2, page 15]
> The term "storage object" is defined as:
> "An object that supports both read and write accesses." [2, page 116]

The intent of the object reuse requirement is stated in the second sentence of the requirement above ("No information . . . produced by a prior subject's actions is to be available to any subject that obtains access to an object that has been released back to the system.") Rephrased, the purpose is to prevent a user who is allocated storage from accessing information put there by a previous user (ignoring, for the moment, explicit sharing). It also has the side effect of preventing a user from accessing information he puts into a storage object which is released to the system and then reallocated to him.

The requirement does not specify how to perform object reuse. It allows the designer some leeway in implementation. An obvious approach is to clear the storage, either upon deallocation or at allocation. Either method is acceptable in meeting the requirement, although there are tradeoffs to consider when making this choice. These issues are discussed later in this guideline. However, the designer can use an approach which, in effect, guarantees that the current "owner' of the storage can only access what he has written (e.g., preventing the reading beyond the end-of-file mark on a tape or in a disk block). Thus, the wording of the requirement recognizes the existence of different implementation strategies which accomplish the same end.

The first sentence of the requirement ("All authorizations to the information contained within a storage object shall be revoked....") means that once a storage object is given to a user, no subject can have access to the information which had been (or may still be) in that object. "All authorizations to the information contained within a storage object shall be revoked...." does not equate to "no subject can have access to that object's contents."Nor does it equate to revoking the authorization attributes (e.g., read, write, execute or append) assigned to information within a storage object. Why? Because if the vendor chooses to clear the storage object "prior to initial assignment, allocation or reallocation," he has revoked "all authorizations to the information contained within [the] storage object." The focus is on the information ---not the storage object, not the attributes.

Does object reuse require the revocation of authorization attributes (e.g., read, write, execute or append) assigned to information within a storage object before it is assigned or allocated? It depends upon the particular implementation. Suppose a system uses memory segments with read, write, execute, and append attributes encoded into descriptors. If the information in the segment has been overwritten, there is no need to clear the descriptor unless the descriptor is considered part of the storage object or is itself a storage object which is released to the system. If the descriptor is considered a storage object or part of one, the attributes are information subject to the object reuse requirement. In both cases, the system could overwrite the old attributes with the new user's attributes. Or it could remove all attributes at deallocation. Either way, clearing the attributes only assures that granting access to a pooled storage object (in this case, a segment) does not implicitly give access to any data previously contained in the object. The information within the storage object must still be protected from the new owner.

Consider the case when a parent process initiates one or more subordinate processes, each having access to a file owned by the parent. Suppose the parent process deletes the file, and the file space (disk space, for example) is returned to the system's resource pool. Does proper object reuse also require the removal of the subordinate processes' access permissions to the file space upon deallocation? No, the requirement allows the system to leave the access set to the parent and/or the offspring processes, but only tO those proCesses. This guarantees that no other processes can access the information in the workspace until the proper procedures are followed. The requirement does not say "revoke authorizations at deallocation of storage objects" ---the requirement does not care if access is continued to be allowed to the process that caused the information to appear in the storage object before the object is reassigned. In other words, a subject could retain effective access to an object that had been returned to the "free pool" until that object was reallocated. (Note: We can think of no reason for, nor an actual implementation of, the above approach. We mention it only to make a point.)

Strictly speaking, the TCSEC object reuse requirement only applies to storage objects accessible by untrusted users of an AIS. However, the system designer may choose to ensure that internal Security relevant data structures are also properly cleared when reusing the internal object. This internal data structure clearing may occur via explicit action of the Trusted Computing Base (TCB), or it may be implicit via a complete overwrite with new TCB data.

An example which illustrates these issues is shown using Table 1. In this example, a segmented memory system provides for virtual storage to system users, using a segment map table (SMT) to maintain segment control information. Control of access to the storage object (i.e., memory segment) is effected by setting the appropriate access attributes to 1 (access permitted) or 0 (access not permitted). Thus, an SMT description pertaining to a segment allocated to subject 51 for read/write access would reflect an attribute coding of R = 1, W = 1, IE = 0, and A = 0 (or 1100).

To satisfy the TCSEC object rouse requirement, the TCB must perform several different actions with respect to the segment and the SMT. The TCB must ensure that the segment is accessible only through known, TCB controlled, mechanisms such as the SMT. This assures the invocation of the TCB whenever an access decision must be made. The TCB must not permit access (by a new subject) to the segment until the segment is specifically allocated to a subject. The TCB must ensure the segment is clear (i.e., contains no residual data) either by explicitly clearing the segment at allocation, clearing the segment at deallocation, or completely over- writing the segment with new data which the subject assigned the segment is permitted to access in the assigned mode. The TCB must also ensure the clearing of any residual control information (i.e., internal TCB data structure) before reallocating the segment (if the control information is contained within

Table 1: SEGMENT MAP TABLE DESCRIPTION

Resident	Secondary		Attribute				Real
Bit	Storage	Length					Address
Address			R	W	E	A	

a storage object). For example, if SMTs are considered storage objects, the TCB must ensure the clearing of any secondary SMT entries for a segment.

2.2 PURPOSE OF THE OBJECT REUSE REQUIREMENT

As stated in section 1.3, the object reuse requirement is primarily designed to satisfy DoD Directive 5200.28 which requires that "Classified information and sensitive unclassified information shall be safeguarded at all times while in AIS's. Safeguards shall be applied so that such information is accessed only by authorized persons . . . .,` [4, page 3] Without this requirement, storage objects could become an information transfer channel between disjoint users as the system reassigns the objects upon user request. Assurance must be provided that no data in any storage object is accessible upon allocation of that object---even if the object is reallocated to the subject having had the most recent access right to it.

2.3 APPLICABILITY OF THE OBJECT REUSE REQUIREMENT

The TCSEC object reuse requirement applies to all storage objects defined by an AIS that are protected by its TCB. The set of storage objects often referred to by a user may be (and likely are) different from the set of physical objects that the system must apply the object reuse requirement to. Whereas the user tends to focus on files or memory space, for example, the system implements these abstract or user-visible objects through the use of real objects such as disk blocks, memory pages or segments, records, bytes, and bits. When a user deletes a file, the system must translate the deallocation of the virtualized file into the deallocation of the corresponding control and data blocks that actually implement the file. A consequence of clearing the physical structures of residual data must be the clearing of the abstract object. Some common physical objects include those items listed in Table 2.

CPU Registers
Floating Point Co-processor Registers
Cache Memory
Physical Memory
Disk Blocks/Sectors
Magnetic Tape Blocks/Sectors
Floppy Disk

Table 2: Common Physical Objects

Storage objects are virtualized into objects that are visible to system users. Some common user objects are discussed in the following section, along with a description of the storage objects used to implement the object.

User Object - Object Implementation

Virtual Memory - Virtual memory is implemented using structures similar to the SMT shown in Figure 1, and physical memory segments and pages. The TCB maintains the structures defining the allocation of physical memory to users. Virtual segments and pages correspond to the physical segments and pages when the virtual memory is mapped into physical memory. Otherwise, the virtual memory resides in a temporary storage area of a physical disk drive or other secondary media.

Files - Files are typically an abstraction of virtual memory and disk blocks or sectors. When a file is not being accessed, it is commonly stored on a physical disk drive. The file is defined by a control block which stores the security critical information of the file and identifies the data blocks corresponding to the file. The control and data blocks are physically resident on the disk drive and are maintained by the TCB. When the file is being accessed, all or part of the file is mapped into the virtual address space (virtual memory) of the process (subject) accessing the file.

Directories - Directories are a special type of file that contains information regarding files. Directories reflect the names of files, and upon referring to a specific filename, the directory provides information to the TCB regarding the physical disk location of the file control block.

Virtual Tape Drives - Tapes drives are generally available to users in one of two fashions: allocating the entire tape drive to a user, or providing access to the tape drive through files (as if the tape drive were a disk drive). When a tape drive is completely allocated to a user, the user can utilize the tape drive in whatever manner desired, so long as the system security policy is adhered to. When a tape drive is accessed through files, the TCB maintains the information relevant to properly protect the files from unauthorized access. This requires the TCB to maintain control and data block information, just as is maintained for files on disk drives.

The object reuse requirement applies to the storage objects defined on an AIS system. But, as is seen in the previous discussion regarding the implementation of virtual objects, it is essential to consider completely what storage objects are used to define the system's objects. By carefully considering this visible object - storage object correspondence the system designer ensures the fulfillment of the object reuse requirement in the AIS system.

Reference [5] proposes a reasonable methodology in conducting an object reuse study. This process is paraphrased here as a representative guide of how one might approach the problem during system development.

Step 1: Identify system objects by focusing on the system documentation that completely describes the TCB interface. This will contain the vast majority of the needed information. If a model exists for the system's security policy, the objects should be contained within it. If not, the remaining system documentation and its architecture should be useful in developing a list of object types.

Step 2: Determine what system level objects are used to create the user-visible objects identified in step 1 above. This is the critical step in the methodology and may represent the greatest amount of analysis. Done correctly, this step determines the bounds of the study. The analysis requires an indepth understanding of the specific operating system involved and knowledge of exactly how user objects are represented.

Step 3: For each identified component, determine how the system initially creates, recognizes, allocates, and deallocates it and returns it for system use. Inherent in this procedure is the need to verify that the requirements originally stated for object reuse processing are met for initialization/ allocation or allocation/deallocation, and that the two are equal. The remaining steps provide this assurance.

Step 4: Examine each system object's initialization routine to ensure that the requirements of object reuse are being met when an object is added to the system. That is, ensure it contains no residual information.

Step 5: Examine the allocation/deallocation routines to ensure that the requirement to remove residual information has been met.

Step 6: Identify all the system operations used to allocate/deallocate objects and ensure that each invokes the object reuse mechanisms necessary to provide the required protection listed in steps three through five above. (Note ---do not forget operations that "only" extend, shorten, or move objects, as these are also allocation/deallocation operations worthy of scrutiny.)

Step 7: Ensure that the free pool of objects itself is protected so that unpurged objects cannot be accessed and read prior to their reallocation. If the system chooses to implement object reuse upon reallocation, there may be a lengthy period of vulnerability where the object with residual information is stored in the free pool. The free pool must guarantee protection of the information so it remains secure until purged prior to reallocation.

Object reuse is essential for all storage objects available in an AS, but an examination of those objects listed in Table 2 reveals that not all such objects are always under TCB control. For example, the TCB can completely control the ability to access data stored in physical memory but cannot protect data stored on magnetic tapes and floppy disks which have been removed from the system. These removable media are subject to mistakes made by operators, and the media may be removed by system users and accessed using off-site equipment. In either of these cases, the TCB does not protect the data stored on the media and must rely to some extent upon physical, procedural, or other controls to ensure that data is not inappropriately accessed or altered. These controls must be discussed in the Trusted Facility Manual for the system. These classes of objects are depicted in Figure 2, where the cache and main memory is normally always under TCB control, the non- removable (fixed) media is usually under TCB control, and removable media is often not under TCB control.

2.4 SUMMARY

Object reuse mechanisms ensure system resources are allocated and reassigned among authorized AIS users in a way which prevents the leak of sensitive information. Object reuse does not necessarily protect against physical attacks on the storage medium. Without specifying how to do it, the TCSEC object reuse requirement applies to classes C2 through A1, with increasing assurance of proper execution as the class increases. The requirement concerns storage objects accessible by untrusted users of an AS; it covers all TCB protected storage objects of an AIS. However, the system designer may clear internal security relevant data structures also. The system must translate the deallocation of virtualized files into the deallocaton of the corresponding control and data blocks actually implementing the files. A result of clearing the physical structures is the clearing of the abstract object. All storage objects require object reuse, but all objects are not always under TCB control. Storage objects may or may not correspond to the objects visible to users. Additionally, some storage objects may need administrative, procedural support to ensure the performance of object reuse.

Figure 2: STORAGE OBJECT CONTROL

Removable Media
Fixed Media
Main Memory - Real & Virtual
Cache Memory

Fully Under TCB Control
Usually Under TCB Control
Often Not Under TCB Control

3 IMPLEMENTATION CONSIDERATIONS FOR OBJECT REUSE

3.1 CONSIDERATIONS FOR PRIMARY STORAGE OBJECTS

Automated Information Systems (ASS) rely upon a collection of primary memory to support the execution of programs. Object reuse on primary storage objects is under the control of the AIS's TCB. The TCB is directly responsible for allocating the primary storage objects, placing data into the objects, and determining when to perform object reuse. The physical memory provides virtual memory for user program execution. When the physical memory is allocated to a user, the TCB must ensure that the memory page contains no residual data. This can be done by purging the memory page with an overwriting technique. For efficiency purposes the designer may wish to use a procedure that overwrites and destroys residual data with the new data being swapped in on behalf of a new subject (in lieu of overwriting with a fixed or random pattern of bits). An important part of this technique is for the system designer to demonstrate that, in all circumstances, the residual data is overwritten prior to the allocation of memory to the incoming subject. Since page frames in main memory are of fixed size and data being swapped in or out is not likely to be an exact fit, it is incumbent upon the designer to ensure all physical page frame space is overwritten (e.g., using a fixed or random pattern of zeros and/or ones to purge the memory from the end of the data being swapped in to the end of the physical page frame).

One must not forget to handle registers. A common error made by system designers is to forget to clear registers that are new to the product or infrequently used by system programmers (e.g., floating point co-processor registers).

Cache memory may present a slightly different set of considerations. If a file is deleted by its owner and portions of the file are retained in cache memory, a later request for information contained in the cache may be honored even though the file no longer exists (unless proper design safeguards are established). This is especially true if the second request occurs very soon after file deletion.

When primary memory is removed from the system (e.g., for maintenance or repair), it is subject to the same considerations discussed later for removable media.

3.2 FIXED MEDIA METHODS

In addition to primary storage objects, many AISs rely on fixed media to contain data for temporary (e.g., buffer or inactive page frame) or long term storage (e.g., file retention). This media may be a fixed disk, the memory in a terminal, and so on. The system designer must identify all physical objects (i.e., blocks, bytes, words) that reside on this media which are used to represent the more abstract objects the user considers (i.e., files, records, programs).

Data is typically stored and retrieved from these kinds of media through the use of several control mechanisms ---each of which might be a storage object in its own right. For example, if a fixed disk or drum is used to provide a backing store capability for a paging system, there are likely several objects associated with the fixed media that will have object reuse requirements. The physical data blocks themselves must be paged between usages, but in addition to this obvious requirement, one must also consider purging the associated entry for the page in the system's page management table and also purging directory information that resides on the media itself.

In all cases, the TCB's ability to fully assure object reuse processing must be demonstrated. This is normally accomplished as discussed earlier in this guideline, through the identification of all objects (and their components) and an examination of each of the routines that control initialization, allocation, and deallocation. As with primary memory, fixed media which is removed from the system is subject to the same considerations discussed later for removable media.

To ensure the system is started in an initially-secure state, the fixed reusable medium must be initialized to an appropriate condition. Clearly, the extent Of the initialization depends on the type of fixed storage objects involved: main memory storage units must be initialized each time a trusted system is started up, while longer-term storage (e.g., disks) would have to be initialized only on first startup or perhaps as part of a recovery process after a system failure.

One can design systems and their attached devices (printers, terminals, etc.) that have residual memory to be cleared with a command issued by the main processor's TCB. However, for open architectures, there is little that can be done from a TCB to (re)initialize the equipment for reallocation. One technique that works with some equipment is to power it off, then on. However, such practice is hard on the equipment and is sometimes not possible without modifying the equipment, or providing TCB-controlled power (outlets) for the equipment. The more likely approach for connected terminal equipment having its own processing capability is to ensure the terminal device itself enforces the object reuse requirement.

3.3 REMOVABLE MEDIA METHODS

This section discusses some possible implementation approaches that could be the basis for an object reuse mechanism for removable (or removed) media. Removable media represents a special case because it can physically be removed from the system. This means that it no longer comes under TCB controls and thus the automated object reuse procedures must be augmented by physical and procedural controls described in the Trusted Facility Manual.

First, any of the methods used for fixed medium systems can be used. As in the fixed storage class, it is important to overwrite in complete physical allocation units (e.g., sectors, clusters, blocks, tracks, cylinders). The actual allocation units are operating system dependent.

As an example, one logical method may apply to tape storage. If the system being rated has no 1/0 commands accessible to users that can force the reading of a tape beyond an end-of-file (IEOF) mark, the TCB can write an EOF mark at the beginning of all `scratch' tapes. Of course, the question is what informs the TCB that a scratch tape is being introduced to the system. As will be seen below, this is a specific instance of a more general question of how to reintroduce removable media to the control of the TCB.

In general, overwriting storage is feasible if the amount of storage to be overwritten is relatively small, and the overwriting is relatively fast. Thus, for most `main memory' usage, overwriting is feasible. It is also feasible for diskettes and other small removable disk media, such as disk cartridges, especially if a multiprogrammed daemon owned by the TCB performs the overwriting.

If real-time overwriting is not feasible (a case for most large removable storage media), then some off-line method is indicated. Only two physical methods appear suitable: degaussing (demagnetizing) and independent off-line overwriting. This guideline cannot recommend any specific degausser, although Government-approved degaussers are available. One commercially available model can completely degauss up to 10 tapes, or a removable disk, in one operation (which takes less than 30 seconds). Degaussers that have been approved to degauss classified tapes or disks are listed on the NSA Degausser Products List [3]. The actual procedures used for degaussing are beyond the scope of this guideline because they again are not processes a TCB can control.

On-line overwriting is a method the TCB could control, possibly by not releasing a tape for removal or reallocation until its entire length is overwritten. However, the performance impact of using on-line overwrite alone is so onerous that it is not a practical solution. Perhaps an efficient method of handling off-line overwrite is to dedicate a small computer system to that function, with one or more tape or disk drives to hold the medium being prepared for reuse. If off-line overwriting is done, there is a lingering concern over whether the overwrite is complete. Even if off-line overwriting is an extension of the object reuse mechanism, one may have to verify the overwrite mechanism always overwrote the entire medium.

3.4 MANAGEMENT OF REMOVABLE MEDIA

The problem with off-line object reuse preparation is not with the method used to eliminate the residue from previous use, but the management of the reintroduction to the TCB of media that was degaussed or overwritten. From the TCB's perspective, it must know it is allocating an object (storage medium) that has been properly processed for reuse. In the case of the fixed media, the TCB itself largely has total control of the process, and the design verification process assures that the TCB only allocates from properly processed reusable objects.

With removable media, the TCB must be informed in some manner that a new or reprocessed (degaussed or off-line overwritten) storage object is being introduced into the allocable storage object pool. Further, since there may be an arbitrary time lapse between a medium being accepted for reuse and its actual reuse, the TCB needs some way to identify removable storage objects that have been accepted for reuse. This is certainly a candidate for incorporation into the system's Trusted Facility Manual.

Clearly, something like the ". . . mechanism by which the TCB and an authorized user reliably communicate . . "to designate the single level of information being imported or exported via single-level communications or I/0 devices could be adapted to allow an operator to reliably inform the TCB that a new or reprocessed removable medium is being reintroduced into the pool of allocable storage objects. At the lower classes the mechanism might be the operator's console. At higher level classes, it might be a variant on the Trusted Path mechanisms between users and the TCB.

How the TCB identifies the removable storage object as being accepted for reuse is again a detail that is very system-dependent. It could range from TCB readable physical identification numbers recorded on the medium being entered into a TCB-managed and -controlled inventory file, to the TCB magnetically (or optically) recording on the medium an encrypted serial number which it also keeps in the inventory of available allocable storage objects.

If degaussing is used as the off-line process to prepare a removable storage object for reuse, it is likely the operating system must prepare it for use by writing timing tracks or sector addresses and by performing other low-level formatting of the medium without which the medium can not be used by the system. This may be sufficient assurance that the TCB has a chance to identify the storage object. However, the TCB must still maintain the inventory of allocable storage objects and assure that only previously accepted storage objects are allocated. As a consequence, it is recommended that formatting of degaussed media is done as a separate function not associated with reentering the medium into the pool of allocable storage objects, and the acceptance and identification process be completely separate from any media preparation required by the system.

While there is no requirement to apply such a technique to all removable media, the notion of economy of mechanism suggests it would be easier to understand if all removable reusable storage is handled in the same way.

3.5 SUMMARY

An AIS's TCB controls object reuse on storage media (primary storage, fixed media, and removable media). Demonstrate the TCB's ability to assure proper object reuse by identifying all physical objects on this media which represent abstract user objects and by examining the routines controlling initialization, allocation, and deallocation. To ensure an initially-secure state for the system, initialize the media to an appropriate condition. Physical and procedural controls augment the automated object reuse procedures of storage media (even primary memory and fixed media) removed from the system and, thus, from TCB controls.

If real-time overwriting of removable storage is not practical, overwrite it off-line or degauss it. Dedicating a small computer system to overwrite storage off-line is an efficient method. When degaussing is used, do not connect media formatting with reentering the media into the pool of allocable storage objects. Separate the acceptance and identification process from media preparation. Handle all removable storage the same way. Incorporate this into the system's Trusted Facility Manual.

Managing the reintroduction of media to the TCB is a problem with off-line object reuse preparation. The TCB must know it is ailocating a properly processed object. For fixed media, the TCB largely controls the process, and design verification assures that the TCB allocates properly processed objects. With removable media, the TCB is told when new or reprocessed storage objects are put into the storage object pool. How this is done is very system-dependent.

GLOSSARY

ASSURANCE

A measure of confidence that the security features and architecture of an automated information system accurately mediate and enforce the security policy.

AUTHORIZATION

The granting of access rights to a user, program, or process.

BROWSING

The act of searching through storage to locate or acquire information without necessarily knowing of the existence or the format of the information being sought.

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, programs, bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, and network nodes.

SCAVENGING

Searching through residue (i.e., objects and/or their components) to acquire unauthorized data.

STORAGE ELEMENT

The hardware (e.g., main memory, disk sectors, magnetic tape) used to store data that is accessible to a user's programs.

STORAGE OBJECT

An object that supports both read and write accesses.

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.

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.1 a user's clearance level) related to the security policy.

REFERENCES

1. A Guide to Understanding Data Remanence in Automated Information Systems, NCSC-TG-025, National Computer Security Center, September 1991.
2. Department of Defense Trusted Computer System Evaluation Criteria, DoD 5200.28-STD, National Computer Security Center, December 1985.
3. "NSA Degausser Products List," Information Systems Security Products and Services Catalogue, National Security Agency, published quarterly.
4. Security Requirements for Automated Information Systems (AISs), DoD Directive 5200.28, Department of Defense, 21 March 1988.
5. Wichers, David R., "Conducting an Object Reuse Study," Proceedings of the 13th National Computer Security Conference, VoI. II, pp. 738-747.