*Proc. of 6th Annual Conference on Computer Assurance (COMPASS 91), Gaithersburg, Maryland, June 1991, pages 63-68.*

# A Case-Study of Security Policy for Manual and Automated Systems

*Edgar H. Sibley, James B. Michael and Ravi S. Sandhu*

Center for Secure Information Systems
and
Department of Information and Software Systems Engineering
George Mason University, Fairfax, VA 22030-4444

**Abstract**

The assumptions underlying security policy models, such as the Bell-LaPadula model, are usually not explicitly stated. This has led to several common misconceptions about this model and its scope. A comparison of manual and automated systems is used to demonstrate the derivation of the Bell-LaPadula $\star$-property (pronounced star-property) for automated systems and its analog for manual systems. This exercise aids in producing a policy model based on needs and a perspective on the limitations of "classical" security policy models.

## 1 Introduction

Many of the assumptions underlying the development of a security policy for information systems are hidden or implicit. For example, the Fundamental Security Policy is seldom, if ever, given explicitly. It may be stated as follows.

**Security Policy 0** *There are objects whose contents should only be disclosed to or changed by authorized people, and these people should not disclose the details (or maybe even the existence) of an object to others who are not authorized to work with the objects. The objects should be preserved over time, if necessary for satisfactory continued operation.*

Due to lack of clarity of underlying assumptions, each user of a security model makes his or her own interpretation of it. This poses a problem for the computer security community. The Bell-LaPadula (BLP) model [1] is the foundation for the U.S. Department of Defense (DoD) Trusted Systems Evaluation Criteria (TCSEC) (also known as the "Orange Book"). For high assurance systems the TCSEC requires derivation of a basic security theorem. Unfortunately, this basic security theorem is based on an incomplete model of the "real world." For example, one of the important assumptions underlying the BLP model is that it only deals with subjects, which are defined as processes acting for or on behalf of the human users and carrying their authority (access rights). This may be sensible if we remember that the model was developed in an operating systems context (i.e., one of interactions between processes and not really including its human users). Many people who rely on the BLP model, however, tend to assume that the model deals with human users as well as subjects.

Other important computer security issues in modern business and organizational environments (such as database applications) cannot be adequately expressed within the framework of BLP and the Orange Book. For instance it is often assumed that data integrity is covered by the BLP model, but in actuality it is not. The BLP model and the Orange Book are only concerned with the integrity of the label on a file, and not of the data contained in the file. Moreover the operating system concept of a file containing only information with one level of classification does not readily accommodate real information systems which contain files with data at different security classifications. There can also be different requirements for integrity in the same file. Also in a business environment, there are restrictions on sharing based on ownership and origin of data which are not expressible within BLP.

Our central theme is that predisposed assumptions in security policy models can leave holes in the security aspects of the information systems that are based on them. In particular information systems based only on the BLP model pose potential problems by allowing new threats to be built in them because the policies are incomplete. In other words there is no policy for handling certain types of threats to such "secure" computer systems. This should not be taken to imply that the BLP specifiers and system implementors were careless, but rather that some needs were deliberately not addressed in the BLP definition. Consequently the needed controls were not built into systems based on BLP.

Here, we propose that a safe—and therefore secure—information system can only be built if the assumptions underlying its security model are made explicit, so that potential hazards (in this case security threats) can be identified and security policies can be formulated (i.e., added to the model) to deal with the potential threats. We consider systems with security requirements to be safety-critical systems. We have adopted the definition of safety-critical software provided in the IEEE Working Draft Standard for Software Safety Plans [2]. According to the standard, an accident is "an unplanned event or series of events that results in death, injury, illness, environmental damage, or damage to or loss of equipment or property. ... [Risk is] a measure that combines both the likelihood that a system hazard will cause an accident and the severity of that accident. ... [Safety-critical software is] software, whose use in a system, can result in unacceptable risk. Safety-critical software includes software whose operation or failure to operate can lead to a hazardous state, software intended to recover from hazardous states, and software intended to mitigate the severity of an accident."

We demonstrate by means of a case study that security policies for manual systems differ from those of automated systems. Our examples are based on potential threats that may occur in each type of system. In this paper, we concentrate mainly on threats to secrecy. We use the terms threat and hazard as synonyms, and define an accident from a security perspective to be an unintentional leakage of information by a user or subject to an unauthorized user or subject.

The rest of this paper is organized as follows. Section 2 presents an example of security policies in a manually operated document library system. It is shown that these policies lead to an analog of the $\star$-property (pronounced star-property) of BLP. Section 3 discusses how the policies of the previous changes require changes or enhancements when applied to an automated system. Section 4 considers some possible extensions to our example. Section 5 concludes the paper.

# 2 An Example: A Secure Manually Operated Document Library System

Our model of a document library system, as illustrated in Figure 1, consists of sets of classified documents serviced by a librarian for users, with messengers to carry the documents and containers for shipping them.

There are two security classifications: blue and red, where blue dominates red. At present, the library does not deal with any special security categories; i.e., there is no current need for the addition of a modifier to the security classification such as "Engineering Department Use Only." Addition of such categories would not entail a major change in the security policy.

In general, the use of categories or compartments means that the ordering of the security labels forms a lattice rather than a total order. The main implication of this generality for our policy statements is that the notion of a "highest" level needs to be replaced by "least upper bound." For simplicity our policies have been stated assuming a total ordering of security classifications.

## 2.1 Security Policy

Because the policy for the manual system is very similar to that of the automated system in the next section, the policy statements in this section have been slightly modified to include the nomenclature used in the Orange Book. In this terminology a subject is a process acting on behalf of a user and objects are the data and files. Thus the statements use terms such as "document (an object)" to allow us to describe the manual system, while also allowing the corollary to these policies to be stated in similar terms for the automated version of the system. For the purpose of this section the parenthetical additions in the policy statements should be ignored. Conversely, for purpose of the next section the parenthetical additions should be substituted for the preceding item.

### 2.1.1 Documents

We first state policy requirements with respect to documents. Each document may have one or more sections (parts) which are classified at different levels. The highest level of any part gives the level of the whole document. Parts may be extracted at a lower level.

**Security Policy 1** *The classification of a document (an object) is a total ordering.*

**Security Policy 2** *Documents (objects) are the entities of interest; they may be complex having parts with different classification levels. They have a dominant label; i.e., the overall classification of a document (object) is at the highest level of any of its parts.*

**Security Policy 3** *A piece of a document (object) may be extracted or copied for use at a lower level of classification.*

### 2.1.2 Users

Next we consider users. All users are assigned a clearance level. They are trusted up to (but not above) their clearance level; i.e., it is assumed that they will not intentionally leak that level of information. Users are also constrained in the actions they may perform on documents; these may be selected from: read, write, or own. Read and write have the obvious interpretations. A document is said to be owned by a user as long as the document is checked out to that user.

Users cannot delegate security levels (e.g., a lower-level secretary is not allowed to perform a higher-level task for the manager). It follows that users only examine the document in a room on their own or with people of the same or higher classification level. This leads us to the following statements.

Figure 1: Document Library System ( Please Refer to The Original Source for the Figure 1)

**Security Policy 4** *All people have clearances. Authority to operate on classified documents may be represented as a triple: $\langle$user, report, type of access$\rangle$, where the type of access is nil or one or more of the set: $\{$read, write, own$\}$. Authority to view classified documents cannot be delegated to people at lower-levels.*

**Security Policy 5** *A user (subject) may only read material in a document (object) that is classified at or below his or her (or its) security clearance level.*[*]

**Security Policy 6** *The user (subject) is not to disclose any classified data to another user (subject) of lower security level unless it is only classified at the lower level.*

**Security Policy 7** *A user (subject) is not permitted to change either the document (object) security label or the label on any part of the document.*[†]

Sometimes users may write or produce a document that is at a higher level than their clearance. Of course, the user is not then allowed to read it! This is termed a write-up.

[*]This property has been called *simple-security* in BLP. It is often described as "no read-up."

[†]This is called the *tranquility property* for objects. Note that it is also implied by Security Policy #12. It may be safer to include such redundant requirements rather than striving for minimal irredundant statements of policy.

**Security Policy 8** *An authorized user (subject) may write material into a document (object) that is classified above his or her (or its) security clearance level.*

### 2.1.3 Messengers

Messengers are untrusted: they only serve to carry messages in locked boxes between users and the librarian. The boxes can only be unlocked by trusted users. The outside of the boxes are labeled with the security level of the enclosed material and these labels cannot be altered. This counters the lack of trust in messengers.

**Security Policy 9** *Containers have an unchangeable external label and this states the highest security level of the enclosed material.*

The next statement may seem obvious and hardly worthy of mention in a manual system. It is required for the sake of completeness and becomes even more important in automated systems.

**Security Policy 10** *Actions on documents (objects) can only be initiated by people (via subjects).*

### 2.1.4 Librarian

Finally we consider the role of the librarian. The librarian is trusted to the highest level of classification in the library. He fills document requests from authorized users and reshelves documents when they are returned by messengers working for the users. Only the librarian (acting as the security officer) can change the classification of a document.

**Security Policy 11** *The librarian (security officer/security kernel) serves as the controller of secure access to documents (objects). Only an authorized user (subject) request will be honored.*

**Security Policy 12** *Only the librarian (security officer/security kernel) can downgrade or declassify documents.*

### 2.2 Analysis

As a consequence of the policy requirements enumerated above we can derive the following requirement on users. Users must be instructed to obey this property and are trusted to do so. It is a sufficiently simple requirement so as to give us confidence that our users can actually adhere to it. It is also similar—but not identical—to the automated system $\star$-property of section 3.

**Theorem 1 [Manual System $\star$-property]** No user is allowed to arbitrarily extract material from a higher-level document and insert it in a lower-level document.

**Proof:** The proof is by contradiction. Suppose that users are allowed to arbitrarily extract material from higher-level documents and insert it in a lower-level document. We show that this will lead to a violation of Security Policy #5. Assume that, at a given time $t_0$, all documents are correctly classified and placed in the relevantly labeled library containers. Recall that blue dominates red. Let A and B be users with clearance blue and red respectively. Let I and J be documents with sensitivity blue and red respectively. Now A can obtain documents I and J. Let A extract a subset of blue pages from I, insert them in J and return I and J to the library. Now B can obtain J and read the blue pages in violation of Security Policy #5.

This argument requires that a user is not permitted to change the label on the pages or cover of a document. Note that by Security Policy #7 the user is trusted not to do this, and by Security Policy #12, the Librarian will not allow such changes internally, except when required for special (trusted) declassification. □

### 2.3 Threats

In spite of the policy requirements enumerated above we can still identify some threats to the security of the library system. One type of threat to the library system is a penetration attack. This may be represented by a badly designed library procedure, which does not require correct or sufficient checking. For example, though forbidden by Security Policy #9, a messenger might find a way to put a red label, or even an unclassified marker, on top of the blue one on the container (essentially declassifying it) and then return it to the librarian for reshelving. The next time the mislabeled document is checked out, it may disseminate information to users with red clearance, even though the information is really supposed to be still classified at the blue level. In this case, there has been deliberate violation of good security policy.

Overuse of messengers and excessive use of the librarian's services also provide the manual equivalent of some signalling or covert channels. If one of the messengers is a spy, he or she can note the calls on the librarian or on other messengers and come to some conclusion about the reason for the increased degree of activity, especially if there is collusion between messengers.

## 3 The Example as an Automated System

In an automated library system, some of the functions performed manually are transferred to computer hardware and software. We need to either

- demonstrate that this type of computerization does not make the overall system less secure, or
- to add necessary requirements into the security policy.

For instance, if the retrieval and storage of documents is automated, we want to demonstrate that the hardware and software used to automate these functions are not vulnerable to penetration or other threats.

In considering the Security Policy statements of the previous section, it is obvious that most of them transfer as written (with the substitution of parenthetical terms to conform with the nomenclature of the Orange book).

Thus Security Policies #1, 2, 3, 4, 5, and 6 present little reason for change. The main question occurs in #2 and 3 when asking what a complex document (object) requires when processing multilevel files: Is the entire "file" returned for a request and how are the various parts "marked" if they have different classification levels? In the BLP type of system this problem is ignored (or considered outside the scope of the policy) because the entire retrieved data is classified at one level and only updated at that level. This is one of the reasons that the use and implementation of Database Management Systems (DBMS) in the context of the DoD requirements are not easily defined, and why the various attempts to develop systems and interpretations of the use of DBMS in a multilevel environment has proved so difficult.

Security Policy #7 (and #12 in a sense) is termed the Tranquility Principle for Objects; it can be stated also as: "An object's sensitivity label can be changed only by a trusted subject." This means also that the user cannot change a label on extracted data prior to writing it back to the database/file system. However, a similar principle must be considered for the activities of the user/subject. In the manual system, it was assumed (though not explicitly covered in the policy) that the user was "known" to the librarian (or had some sort of badge, etc.). This means that the user cannot arbitrarily be replaced by a person with a lower clearance.

New types of threats can enter with automation; e.g., a copier could be placed in the library, and when a copy of a document is made by a trusted user, the drum of the machine or a separate piece of paper that remains in the machine may retain another (covert) copy to be retrieved by a corrupted user. In this situation the copier is said to contain a Trojan Horse; i.e., in addition to its normal useful function the copier acts surreptitiously to cause a security flaw. Another example of a Trojan Horse is a text editor which has been modified to incorrectly display the label on a document causing the user to divulge secret information. Both of the above examples of Trojan Horses require improved security policies to specify checks on the activities performed in the library and by its servers.

Indeed, the major difference between the requirement for a manual and an automated system is the number and type of possible threats, which are potentially higher for an automated system. The threats are all those previously considered plus additions for the automated system; e.g., the use of COTS (commercial off-the-shelf software) means that the system incorporates code provided by (untrusted) people outside the organization, while the high speed of the hardware activities allows much faster passage of information over covert channels. Obviously, when an untrusted piece of software is called into service by a user, the system must make sure that there is no way that this subject is masquerading as a higher-level user. This gives rise to two additional policies.

**Security Policy 13** *The user's security classification data, which is used while the authorization of an action is being checked, must be retained in a secure and trusted environment and the programs that access this data must be trusted.*

**Security Policy 14** *A subject (which may be untrusted software) takes on the authority (sensitivity label) of the initiating user. It is essential that this label can be changed only by a trusted subject.*[‡]

In the automated world, certain researchers and implementors consider Security Policy #8 an anathema. They ask why it is necessary and can show that it introduces significant problems in DBMS applications. Naturally, Security Policies #9 and 10 are essentially augmented by #13 and 14 above.

Unfortunately, because the automated servers cannot be trusted, there is a need for a trusted subsystem that takes on the controls and checks of the librarian (in Security Policy #11): giving rise to the "trusted kernel," with various possible levels of confidence with respect to the Orange Book (from the relatively low-level but normal commercial needs to a requirement that it be formally specified and its specification verified). Security Policy #12, of course still defines one of the security officer's functions.

Finally the analog to Theorem 1 is given below.

**Theorem 2 [Man-Machine System $\star$-Property]** An (untrusted) subject who has read an object at level l is not allowed to write to objects classified at levels strictly below l.

**Proof:** By the same arguments as for theorem 1, no subject (trusted or untrusted) is allowed to arbitrarily extract material from a higher-level document and insert it in a lower-level document. By definition untrusted subjects exhibit unknown and perhaps devious behavior. Therefore the requirement of theorem 1 can be enforced for such subjects only by the stipulation given in this theorem. $\square$

The statement of this theorem is conditioned on what the subject has read. In practice the slightly stronger requirement—that a subject cannot write below its level—is usually enforced. This requirement is usually called "no write-down."

## 4 Possible Extensions

Possible extensions to the types of threats discussed in the previous sections include those of spies and the inference problem. We define a spy to be a corrupted user. Spies can exist in both manual and automated systems, even if they run at system-high.[§] There are no policies specified for dealing with spies or other untrusted subjects. Physical security policies (e.g., requiring the use of available protective devices or human security) need to be explicitly specified. Enumerating all of the possible types of attacks is an impossible problem. We assume that there are an infinite number of possible attacks that could be used by a corrupted user. There is, however, probably a subset of these attacks that are more likely to be used by particular types of corrupted users and/or against particular types of secure information systems. In the latter case, the problem may be bounded.

Inference and aggregation allow leakage of information to untrusted or unauthorized users or subjects via multiple accesses or the aggregation of information from low classification levels. It is difficult to prevent the use of inference in databases, because its instantiation is not known *a priori* by the system architects, who must state the policies for accessing and combining information at different classification levels. Moreover, the database schema or classification criteria may change over time.

The inference problem could occur in our library example if a user was able to piece together blue information from a set of red documents or by asking a

[‡]This is the tranquility principle for subjects.

[§]A system is said to run at system-high if all users are cleared to the level of the data with the highest classification.

"reference librarian" questions about various topics or documents.

In addition to these threats, there are many unknowns (until they are detected) and threats may even cause leakage of information that is undetected for some time.

## 5 Conclusions

To summarize, we have compared security policies in a manual and an automated document library system. We have described some of the security threats posed in the manual and automated versions of the system.

We are currently developing a policy workbench. This will be used to perform experiments in formalizing security policies and proving theorems about formal security policy models working with the security policies as axioms. Our ultimate goal is to provide security officers with a set of integrated tools for representing and analyzing security policy models and the degree of security in the information system architectures (e.g., How much assurance is required? What are the people policy issues? What are the tradeoffs involved in achieving a particular level of security?).

One research issue of particular interest is: What impact do changes in technology have on the policy-to-model transformation process? An example of the impact of technology on security policy and models is in the increase in the number of possible covert channels as advances in hardware architectures provide additional computing resources (e.g., communication and processing performance). There is a tradeoff between system performance and the methods (i.e., procedures that implement policy) used to counter covert channels (e.g., sending random signals across the network).

## References


[1] Bell, D. E., and LaPadula, L. J. Secure Computer System: Unified Exposition and Multics Interpretation. Technical Report MTR-2997, Mitre Corporation, March 1976.

[2] U.S. Department of Defense. Trusted Computer System Evaluation Criteria. DoD 5200.28-STD, December 1985.

[3] Institute for Electrical and Electronics Engineers. Standard for Software Safety Plans. IEEE Software Safety Working Group P1228, Working Draft D, March 1991.