SECURE COMPUTER SYSTEMS: MATHEMATICAL FOUNDATIONS

D. Elliott Bell, et al

Mitre Corporation

Prepared for:
Electronics Systems Division

November 1973
When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

Publication of this technical report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.

MELVIN B. EMMONS, Colonel, USAF
Director, Information Systems Technology
Deputy for Command & Management Systems
SECURE COMPUTER SYSTEMS: MATHEMATICAL FOUNDATIONS

D. E. Bell, L. J. LaPadula

The MITRE Corporation
Box 206
Bedford, Mass. 01730

Deputy for Command and Management Systems
Electronics Systems Division, AFSC
L. G. Hanscom Field, Bedford, Mass. 01730

Approved for public release; distribution unlimited.

This paper reports the first results of an investigation into solutions to problems of security in computer systems; it establishes the basis for rigorous investigation by providing a general descriptive model of a computer system.

Borrowing basic concepts and constructs from general systems theory, we present...
a basic result concerning security in computer systems, using precise notions of "security" and "compromise". We also demonstrate how a change in requirements can be reflected in the resulting mathematical model.

A lengthy introductory section is included in order to bridge the gap between general systems theory and practical problem solving.
This is Volume I of a multi-volume report prepared by The MITRE Corporation, Bedford, Massachusetts, in support of Project 522B under Contract No. F19628-73-C-0001.

The authors of the report are D. Elliott Bell and Leonard J. LaPadula of the MITRE Corporation.

This report represents an initial attempt at specifying requirements for a secure computer system based upon the development and verification of a mathematical model.

The assumptions and specifications relating to security requirements as expressed in the report are not necessarily applicable to any specific system. The development presented here will help to reveal and clarify the basic problems and issues confronting designers of multi-level secure computer systems.
PREFACE

General systems theory is a relatively new and rapidly growing mathematical discipline which shows great promise for application in the computer sciences. The discipline includes both "general systems-theory" and "general-systems theory": that is, one may properly read the phrase "general systems theory" in both ways.

In this paper, we have borrowed from the works of general systems theorists, principally from the basic work of Mesarovic, to formulate a mathematical framework within which to deal with the problems of secure computer systems. At the present time we feel that the mathematical representation developed herein is adequate to deal with most if not all of the security problems one may wish to pose. In Section III we have given a result which deals with the most trivial of the secure computer systems one might find viable in actual use. In the concluding section we review the application of our mathematical methodology and suggest major areas of concern in the design of a secure system.

The results reported in this paper lay the groundwork for further, more specific investigation into secure computer systems. The investigation will proceed by specializing the elements of the model to represent particular aspects of system design and operation. Such an investigation will be reported in the second volume of this series where we assume a system with centralized access control. A preliminary investigation of distributed access is just beginning; the results of that investigation would be reported in a third volume of the series.
TABLE OF CONTENTS

LIST OF TABLES  Page

SECTION I  INTRODUCTION  Page 1
  GENERAL SYSTEMS  1
  SYSTEM MODELING  6
  SECURE COMPUTER SYSTEMS  9
  PROBLEMS OF SECURITY  10
  SUMMARY AND REFERENCES  10

SECTION II  FOUNDATIONS OF A MATHEMATICAL MODEL  Page 14
  ELEMENTS OF THE MODEL  14
  STATES OF THE SYSTEM  17
  STATE-TRANSITION RELATION  17
  SUMMARY AND REFERENCES  18

SECTION III  A FUNDAMENTAL RESULT  Page 19
  COMPROMISE AND SECURITY  19
  ASSUMPTIONS  20
  BASIC SECURITY THEOREM  21
  SUMMARY  22

SECTION IV  CONCLUSION  Page 23
  INTRODUCTION  23
  PROBLEM REFORMULATION  24
    Basic Security Theorem (revised)  25
    ACCESS CONTROL  27
    DATA BASE SHARING  28
    SUMMARY AND REFERENCES  29

BIBLIOGRAPHY  Page 31
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>State-Transition Table</td>
<td>5</td>
</tr>
<tr>
<td>II</td>
<td>Elements of the Model</td>
<td>14</td>
</tr>
<tr>
<td>III</td>
<td>Initial Requirements</td>
<td>20</td>
</tr>
<tr>
<td>IV</td>
<td>Modified Requirements</td>
<td>25</td>
</tr>
</tbody>
</table>
SECTION I
INTRODUCTION

GENERAL SYSTEMS

We shall begin by presenting a brief description of general systems theory as we shall use it in this paper. We consider a system in its most general form to be a relation on abstract sets. We express this mathematically by the expression

\[ S \subseteq X \times Y \]

where the system \( S \) is a relation on the abstract sets \( X \) and \( Y \). If \( S \) is a function from \( X \) to \( Y \) (\( S: X \rightarrow Y \)), then it is natural to consider \( S \) to be a functional system. In this case, it is convenient to consider the elements of \( X \) to be inputs and the elements of \( Y \) to be outputs so that \( S \) expresses a functional input-output relationship. By appropriate choice of the sets \( X \) and \( Y \) (and a set \( Z \) to represent states when necessary), one can closely represent some situation of particular interest and reach significant conclusions about that situation.

This very general definition of a system provides a framework of investigation which has very wide applicability and, as we shall see in Section III, unexpected power. We shall illustrate the concept's applicability with three examples.

**Example 1**: Consider a savings account in a bank which compounds interest quarterly. The general situation of varying payments, withdrawals, and interest rates can be described by a difference
equation as follows:

\[ b_k = (b_{k-1} + p_k) \cdot (1 + i_k) \]  (1.1)

where \( b_k \) represents the balance after the computation of interest at the end of the \( k \)-th quarter, \( p_k \) represents the net transaction (that is, the net of deposits and withdrawals) in the account during the \( k \)-th quarter, \( \dagger \) and \( i_k \) represents the quarterly interest rate at the end of the \( k \)-th quarter. A seven-year history of such a savings account (seven years for tax purposes) is represented by a system

\[ S(b_0) \subseteq \mathbb{P} \times \mathbb{I} \times \mathbb{B} \]

where

- \( b_0 \) represents the initial balance in the account;
- \( \mathbb{P} = \mathbb{R}^{28} \) represents the twenty-eight transactions;
- \( \mathbb{I} = \mathbb{R}^{28} \) represents the twenty-eight quarterly interest rates;
- \( \mathbb{B} = \mathbb{R}^{28} \) represents the twenty-eight successive balances

and \((p,i,b) \in S(b_0)\) if and only if equation (1.1) holds for every \( k \) from 1 to 28 inclusive, where \( p = (p_1, \ldots, p_{28}) \);
\( i = (i_1, \ldots, i_{28}) \); and \( b = (b_1, \ldots, b_{28}) \). The system \( S(b_0) \) describes in full generality the seven-year savings-account history in any circumstance. Certain results in econometrics are equivalent to determining \( b_{28} \) under further specific assumptions. For example, the determination of \( b_{28} \) for \((p,i,b) \in S(0)\) where \( p_2 = \cdots = p_{28} = 0 \) and \( i_1 = i_2 = \cdots = i_{28} > 0 \) is accomplished using the

\[ \dagger \text{We assume for simplicity that interest is paid on the amount in the account at the end of the quarter.} \]

\[ \dagger \text{The set of 28-tuples of real numbers.} \]
compound interest formula

\[ b_{28} = p_1 \cdot (1 + i_1)^{28}. \]

A number of remarks concerning this example are in order. It is certainly true that the use of an econometric table prepared for a specific situation is easier than the direct use of the difference equation (1.1). On the other hand, small changes in a situation can make the use of tables cumbersome. For example, suppose that the \( p_j \) in the sequence \((p_1, p_2, \ldots, p_{28})\) are positive and distinct and that \( i_1 = i_2 = \cdots = i_{28} > 0 \). Then by use of econometric tables, we compute \( b_{28} \) by the formula

\[ b_{28} = \sum_{j=1}^{28} p_j \cdot (F/P, i_1, 29-j). \]

This means that the compound amount factor \((F/P, i_1, 29-j)\) must be looked up 28 times in the compound interest factors table one is using. If we further complicate the problem by having the \( i_j \) in \((i_1, i_2, \ldots, i_{28})\) distinct and positive, then we could compute \( b_{28} \) by the iterative method:

\[ b_{28} = (b_{27} + p_{28}) \cdot (F/P, i_{28}, 1) \]
\[ b_{27} = (b_{26} + p_{27}) \cdot (F/P, i_{27}, 1) \]
\[ \cdots \]
\[ b_1 = (b_0 + p_1) \cdot (F/P, i_1, 1); \]

or we could use the single formula obtainable by straightforward algebraic substitution in the equations above. So, to find \( b_{28} \),

*See [5], page 594.*
we start with $b_0$ and work backwards; in using the compound interest factors tables we should have to do 28 look-ups, each on a different page since in each quarter the interest is different from that in any other quarter. If it happens that each $i_j < k\%$, where $k\%$ is the lowest interest for which we have a table, our problem has become even more severe. It is much easier in these cases, especially on a digital computer, simply to use the difference equation (1.1).

The preceding remarks should illustrate that the most important characteristics of the system (that is, the difference equation) are its appropriateness to the situation modeled and its general applicability.

Example 2: Consider the motion of a body $B$ suspended on an ideal spring. The motion is governed by the differential equation

$$m \cdot s''(t) + k \cdot s(t) = x(t) \quad (1.2)$$

where $m$ is the mass of $B$, $s(t)$ is the position of $B$ at time $t$, $k$ is a constant of the spring, and $x(t)$ is an external force acting on $B$ at time $t$. If $C$ is the set of all analytic functions on $[0,\infty)$, then the differential equation (1.2) with initial conditions $s(0) = a$ and $s'(0) = b$ is represented by the system $S(a,b)$ defined as follows:

$$S(a,b) \subseteq C \times C$$

where $(x(t), s(t)) \in S(a,b)$ if and only if $s(0) = a$, $s'(0) = b$, and the functions $x$ and $s$ satisfy (1.2) for all $t \in [0,\infty)$. Hence the familiar analytical tool of differential equations is a
system under our very broad definition. Our third example will show that finite-state machines are also encompassed in our concept of system.

Example 3: Consider a vending machine which accepts nickels, dimes, and quarters for a ten-cent cup of coffee and gives change if any is due. Let \( A = \{5, 10, 25\} \) represent the coins acceptable to the machine. Let \( B_1 = \{\emptyset, C\} \) where "\( \emptyset \)" means "no coffee" and "C" means "coffee." Let \( B_2 = \{0, 5, 10, 25\} \) represent the coins the machine can return. The set \( B = B_1 \times B_2 \times B_2 \) specifies the set of outputs that can occur at any time. Now let the set \( Q = \{q_0, q_1\} \) represent the states of the machine. We give a state transition function \( f: A \times Q \rightarrow Q \) and an output function \( g: A \times Q \rightarrow B \) by the following table:

<table>
<thead>
<tr>
<th>( a = 5 )</th>
<th>( a = 10 )</th>
<th>( a = 25 )</th>
<th>( a = 5 )</th>
<th>( a = 10 )</th>
<th>( a = 25 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(a,q_0) )</td>
<td>( q_1 )</td>
<td>( q_0 )</td>
<td>( q_0 )</td>
<td>( (\emptyset, 0, 0) )</td>
<td>( (C, 0, 0) )</td>
</tr>
<tr>
<td>( f(a,q_1) )</td>
<td>( q_0 )</td>
<td>( q_0 )</td>
<td>( q_1 )</td>
<td>( (C, 0, 0) )</td>
<td>( (C, 5, 0) )</td>
</tr>
</tbody>
</table>

We have now modeled the vending machine as a finite-state machine in the usual manner.

Now suppose that we observe \( n \) trials. Let \( A^n \) and \( B^n \) be, respectively, the sets of all \( n \)-tuples from the sets \( A \) and \( B \). Then for a given initial state \( q = q_i \), \( i \subseteq \{0, 1\} \), there corresponds
to any input tape $x$ in $A^n$ a unique output tape $y$ in $B^n$. We have defined a mapping

$$S_q : A^n \rightarrow B^n$$

such that for each $x$ in $A^n$ the image $y = S_q(x)$ is the unique output sequence corresponding to the input sequence $x$ and the initial state $q = q_1$. We say that the vending machine is represented by the system $S \subseteq A^n \times B^n$ where $S = S_{q_0} \cup S_{q_1}$. Considering that in normal operation of the machine the initial state is $q_0$, we can consider the vending machine to be the functional system $S_{q_0}$.

The examples we have presented are intended to enhance the intelligibility of the discussion of system modeling in the next section. Additionally, the enrichment of one's intuitive notions through the use of examples will, hopefully, serve a similar purpose in the next section.

**SYSTEM MODELING**

The mathematics of relations among objects with which we deal is designed to provide a useful model for our investigation of secure computer systems. Three desirable properties of such a model suggested by the examples of the previous section are generality, a predictive ability, and appropriateness. In this section, we shall discuss each of these properties in turn, commenting on its relation to a "useful" model of a particular situation.

Differential equations are systems that frequently display great generality. Equation (1.2) illustrates this point clearly.
Without knowing the mass of \( B \) and without specifying the spring constant \( k \), we can nevertheless analyze the general system. In fact, for \( x(t) = 0 \), (1.2) has the closed form solution

\[
s(t) = A \cdot \sin(nt + C),
\]

where \( n = (k/m)^{1/2} \) and \( A \) and \( C \) are constants determined by the initial conditions \( a \) and \( b \). Moreover, equation (1.2) is a special case of the more general form

\[
s''(t) + 2k \cdot s'(t) + n^2 \cdot s(t) = x(t)
\]

which models a vast number of elastic vibrations including electrical oscillations (as in a capacitor) and the vibrations in pipe organs [2].

A model too closely tied to a specific application loses the possibility of more general applicability. On the other hand, a model insufficiently rooted in the problem at hand will not allow accurate prediction of the behavior of the physical system being modeled. For example, knowing the initial conditions of the suspended weight \( B \), the mass of \( B \), and the spring constant \( d \), we can predict precisely where \( B \) will be 5.83337 seconds from "let-go." The same sort of precise predictive power is desirable in modeling discrete computer systems. Moreover, in modeling secure computer systems we must deny ourselves the luxury of accepting approximate answers and insist on absolute rather than probabilistic determinacy.

The last important feature of a model is its appropriateness to the situation of interest. In each of the three examples of Section I, the type of system used appropriately described the important properties of the situation being modeled. One particular
advantage of an appropriate model can be illustrated by the third example, while the severe problems which an inappropriate model can cause can be demonstrated by a discussion of the second example.

The vending machine modeled in Example 3 illustrates that problems other than correctness can be detected in a model appropriate to a given situation. In particular, the machine we have defined has this interesting characteristic: if in state \( q_1 \) one continually inserts quarters into the machine, the machine monotonously returns a quarter and gives no coffee. This is a behavioral characteristic which the vending machine company might consider undesirable. We have purposely constructed our sample machine in this way in order to show that while the machine is "correct" in its operation, we may consider it to be non-viable as a profit-making item.*

Now consider the situation modeled in Example 2. If a discrete model had been chosen over a continuous one, the model might have been represented by discrete observations of the spring-weight tandem

\[ u_t = s(t), \quad t = 0, 1, 2, 3, \ldots \quad (1.4) \]

where \( s(t) \) is the same position function appearing in (1.2).

Suppose \( B \) has mass = 1 gram, the time interval is 1 second, and the spring constant is \( k = 39.478 \, \text{g/sec}^2 \). In this special case, the motion of \( B \) indicates no apparent movement—the body \( B \) is always the same position \( s(0) \) at each observation time. The

*This characteristic (i.e., returning quarters inserted after a single nickel has been put into the machine) is one which might irritate customers and not sell coffee in the process. An alternative approach which, although not correct, might be more acceptable to a vending machine company would be to set \( f(25, q_1) = q_0 \) and \( g(25, q_1) = (C, 5, 10) \): that is, make change for the quarter, supply coffee, and ignore the nickel. Purposefully or inadvertently, this may well be the course chosen by some vending machine companies.
periodicity of B's motion is precisely what makes a continuous differential-equation model more appropriate than a discrete model of the type described (in addition to the more accurate predictive power). The point is that an inappropriate model of a problem situation can obfuscate the essential issues involved, thus complicating the problem.

The major task in system modeling is to provide a useful model of the situation under scrutiny, a model which exhibits generality, a predictive ability, and appropriateness to the problem at hand.

SECURE COMPUTER SYSTEMS

A number of systems have been built and designed which attack the general problem of security in some form and to some extent. In some cases, privacy of data is the principal objective; in others, the prime objective is access control. For the security criteria which we shall establish, however, no existing system of which we are aware is adequate. *

When we speak of a secure computer system, we mean one which satisfies some definition of "security". Our interest is security in the usual military and governmental senses — that is, security involving classifications and need-to-know.

We shall investigate a bounded form of the general problem of security. Our interest shall be to certify that within the digital computer, which is only part of a total system, no security compromise will occur. The elements with which we shall deal, then, are processes (programs in execution), data, access control algorithms, classifications of data and processes, and the needs-to-know of elements within the digital computer.

*See reference [13] at the end of this section.
PROBLEMS OF SECURITY

Let us consider a security compromise to be unauthorized access to information, where unauthorized means that an inappropriate clearance or a lack of need-to-know is involved in the access to the information. Then a central problem to be solved within the computing system is how to guarantee that unauthorized access (by a process) to information (file, program, data) does not occur.

If we can certify that unauthorized access cannot occur within the system, then we must next consider the secondary effects of the method by which security has been achieved. Principally we shall have to address ourselves to the general question of the viability of the resultant system in terms of economic and technological feasibility and in terms of usefulness to the user.

SUMMARY AND REFERENCES

In this chapter we have introduced general systems theory very briefly and have shown examples of its application. Together with the short discussion on system modeling, the general systems theory and examples should provide an adequate basis for reading the rest of this paper.

The reader who may wish to investigate systems theory for himself is referred first to the book edited by Klir [9], which can profitably be read with or without any background in mathematics. The reader will find further examples of systems in the book [14] by Mesarović, Macko, and Takahara. In particular, beginning on page 69 of [14] the reader will find the basic mathematical concept of a system which we have borrowed. Other books which should be of interest are those by Klir [8], Hamer [6], von Bertalanffy [1], and Zadeh and Polak [15].
In the section entitled SECURE COMPUTER SYSTEMS we defined in broad terms what we mean by a secure computer system. Our general notion of a secure system is derived in large measure from essentials of a secure system abstracted from the Multics system, as an archetype of multi-user systems, and from a knowledge of security problems. The reader can find numerous articles in the literature which touch on the area of a secure computer system; we list [3,4,10,11,12] as representative of what is available. As we pointed out, however, none of the generally available literature deals specifically with the problem we address in this paper.

Finally, we have indicated in this chapter what we consider to be the general problems we shall encounter in investigating secure computer systems.


SECTION II
FOUNDATIONS OF A MATHEMATICAL MODEL

ELEMENTS OF THE MODEL

We begin by identifying elements of the model which correspond to parts of the real system to be modeled. We assume the real system to have multiple users operating concurrently on a common data base with multi-level classification for both users and data and need-to-know categories associated with both users and data. In our model we deal with subjects (processes), which one should consider surrogates for the users.

We show the elements of our model in Table II, wherein we identify sets, elements of the sets, and an interpretation of the elements of the sets.

Table II
Elements of the Model

<table>
<thead>
<tr>
<th>Set</th>
<th>Elements</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>{S_1, S_2, \ldots, S_n}</td>
<td>subjects; processes, programs in execution</td>
</tr>
<tr>
<td>O</td>
<td>{O_1, O_2, \ldots, O_m}</td>
<td>objects; data, files, programs, subjects</td>
</tr>
<tr>
<td>C</td>
<td>{C_1, C_2, \ldots, C_q}</td>
<td>classifications; clearance level of a subject, classification of an object</td>
</tr>
<tr>
<td>C_1 &gt; C_2 &gt; \ldots &gt; C_q</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>{K_1, K_2, \ldots, K_r}</td>
<td>needs-to-know categories; project numbers, access privileges</td>
</tr>
</tbody>
</table>

14
<table>
<thead>
<tr>
<th>Set</th>
<th>Elements</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>${A_1, A_2, \cdots, A_p}$</td>
<td>access attributes; read, write, copy, append, owner, control</td>
</tr>
<tr>
<td>R</td>
<td>${R_1, R_2, \cdots, R_u}$</td>
<td>requests; inputs, commands, requests for access to objects by subjects</td>
</tr>
<tr>
<td>D</td>
<td>${D_1, D_2, \cdots, D_v}$</td>
<td>decisions; outputs, answers, &quot;yes&quot;, &quot;no&quot;, &quot;error&quot;</td>
</tr>
<tr>
<td>T</td>
<td>({1, 2, \cdots, t, \cdots})</td>
<td>indices; elements of the time set; identification of discrete moments; an element (t) is an index to request and decision sequences</td>
</tr>
<tr>
<td>$P_a$</td>
<td>all subsets of (a)</td>
<td>power set of (a)</td>
</tr>
<tr>
<td>$a^\beta$</td>
<td>all functions from the set (\beta) to the set (a)</td>
<td></td>
</tr>
<tr>
<td>$\alpha \times \beta$</td>
<td>${(a, b) : a \in \alpha, b \in \beta}$</td>
<td>Cartesian product of the sets (\alpha) and (\beta)</td>
</tr>
<tr>
<td>$F$</td>
<td>$C^S \times C^O \times (PK)^S \times (PK)^O$</td>
<td>classification/need-to-know vectors; an arbitrary element of $F$ is written $f = (f_1, f_2, f_3, f_4)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f_1$: subject-classification function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f_2$: object-classification function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f_3$: subject-need-to-know function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f_4$: object-need-to-know function</td>
</tr>
</tbody>
</table>

15
<table>
<thead>
<tr>
<th>Set</th>
<th>Element</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>$R^T$</td>
<td>request sequences</td>
</tr>
<tr>
<td></td>
<td>an arbitrary element of $X$ is written $x$</td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>$D^T$</td>
<td>decision sequences</td>
</tr>
<tr>
<td></td>
<td>an arbitrary element of $Y$ is written $y$</td>
<td></td>
</tr>
<tr>
<td>$M$</td>
<td>${M_1, M_2, \ldots, M_{nm^2p}}$</td>
<td>access matrices</td>
</tr>
<tr>
<td></td>
<td>an element $M_k$ of $M$ is an $n \times m$ matrix with entries from $PA$; the $(i,j)$-entry of $M_k$ shows $S_i$'s access attributes relative to $O_j$</td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>$P(S \times O) \times M \times F$</td>
<td>states</td>
</tr>
<tr>
<td>Z</td>
<td>$V^T$</td>
<td>state sequences</td>
</tr>
<tr>
<td></td>
<td>an arbitrary element of $Z$ is written $z$; $z_t \in z$ is the $t$-th state in the state sequence $z$</td>
<td></td>
</tr>
</tbody>
</table>
STATES OF THE SYSTEM

We have defined the states of the system in such a way as to embody all the information which we consider pertinent to security considerations.

A state \( v \in V \) is a 3-tuple \((b, M, f)\) where

- \( b \in P(S \times O) \), indicating which subjects have access to which objects in the state \( v \);
- \( M \in M \), indicating the entries of the access matrix in the state \( v \); and
- \( f \in F \), indicating the clearance level of all subjects, the classification level of all objects, and the needs-to-know associated with all subjects, and objects in the state \( v \).

STATE-TRANSITION RELATION

Let \( W \subseteq R \times D \times V \times V \). The system \( \Sigma(R, D, W, z_0) \subseteq X \times Y \times Z \) is defined by

\[(x, y, z) \in \Sigma(R, D, W, z_0) \text{ if and only if } (x_t, y_t, z_t, z_{t-1}) \in W \text{ for each } t \in T, \text{ where } z_0 \text{ is a specified initial state usually of the form } (\phi, M, f), \text{ where } \phi \text{ denotes the empty set.}\]

\( W \) has been defined as a relation. It can be specialized to be a function, although this is not necessary for the development herein. When considering design questions, however, \( W \) will be a function, specifying next-state and next-output. \( W \) should be considered
intuitively as embodying the rules of operation by which the system in any given state determines its decision for a given request and moves into a next state.

SUMMARY AND REFERENCES

In this section we have established elements of a mathematical model of a system; these elements were chosen to represent as nearly as possible the realities of the problem situation and to enable as easy a transition as possible from mathematical model to design specifications.

The states of the system have been defined in such a way as to incorporate all information which seems pertinent to correct operation of a secure system ("secure system" to be defined precisely in the next section).

Finally, we have included in the model a state-transition relation which is the key to modeling: given W one may predict the behavior of the system for a given set of initial conditions and a given request sequence.
SECTION III
A FUNDAMENTAL RESULT

COMPROMISE AND SECURITY

We define a compromise state as follows: \( v = (b,M,f) \in V \) is a compromise state (compromise) if there is an ordered pair \((S,O) \in b\) such that

(i) \( f_1(S) < f_2(O) \) or
(ii) \( f_3(S) \not< f_4(O) \).

In other words, \( v \) is a compromise if the current allocation of objects to subjects \( b \) includes an assignment \((S,O)\) with at least one of two undesirable characteristics:

(i') \( S \)'s clearance is lower than \( O \)'s classification;
(ii') \( S \) does not have some need-to-know category that is assigned to \( O \).

In order to make later discussions and arguments a little more succinct, we shall define a security condition. \((S,O) \in S \times O\) satisfies the security condition relative to \( f \) (SC rel \( f \)) if

(iii) \( f_1(S) \geq f_2(O) \) and
(iv) \( f_3(S) \supseteq f_4(O) \).

A state \( v = (b,M,f) \in V \) is a secure state if each \((S,O) \in b\) satisfies SC rel \( f \). The definitions of secure states and compromise states indicate the validity of the following unproved proposition.
**Proposition:** \( v \in V \) is not a secure state iff \( v \) is a compromise.

A state sequence \( z \in Z \) has a compromise if \( z_t \) is a compromise for some \( t \in T \). \( z \) is a secure state sequence if \( z_t \) is a secure state for each \( t \in T \). We shall call \((x,y,z) \in \Sigma(R,D,W,z_0)\) an appearance of the system. \((x,y,z) \in \Sigma(R,D,W,z_0)\) is a secure appearance if \( z \) is a secure state sequence. The appearance \((x,y,z)\) has a compromise if \( z \) has a compromise.

\(\Sigma(R,D,W,z_0)\) is a secure system if every appearance of \(\Sigma(R,D,W,z_0)\) is secure. \(\Sigma(R,D,W,z_0)\) has a compromise if any appearance of \(\Sigma(R,D,W,z_0)\) has a compromise.

**Proposition:** \( z \in Z \) is not secure iff \( z \) has a compromise.

**Proposition:** \( \Sigma(R,D,W,z_0) \) is not secure iff \( \Sigma(R,D,W,z_0) \) has a compromise.

**ASSUMPTIONS**

We make assumptions, as shown in Table III, which reflect a subset of requirements (or lack of requirements) to be imposed on the system. In Section IV we shall change some of these assumptions and observe the effect on the system.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Raise?</th>
<th>Lower?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject Clearance</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Object Classification</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Subject Needs-to-Know</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Object Needs-to-Know</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>
Table III, in effect, says that "no" is the answer to each of the questions

<table>
<thead>
<tr>
<th>raise</th>
<th>lower</th>
<th>increase</th>
<th>decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject's classification/clearance needs-to-know</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Basic Security Theorem:

Basic Security Theorem: Let \( W \subseteq R \times D \times V \times V \) be any relation such that \((R_i, D_j, (b^*, M^*, f^*)), (b, M, f) \in W\) implies

1. \( f = f^* \) and
2. every \((S, O) \in b^* - b\) satisfies SC rel \( f^* \).

\( \Sigma(R, D, W, z) \) is a secure system for any secure state \( z \).

Proof: Let \( z_0 = (b, M, f) \) be secure. Pick \((x, y, z) \in \Sigma(R, D, W, z)\) and write \( z_t = (b(t), M(t), f(t)) \) for each \( t \in T \).

\( z_1 \) is a secure state. \((x_1, y_1, z_1, z) \in W\). Thus by (1), \( f^{(1)} = f \).

By (ii), every \((S, O) \in b^{(1)} - b\) satisfies SC rel \( f^{(1)} \). Since \( z \) is secure, every \((S, O) \in b\) satisfies SC rel \( f \). Since \( f = f^{(1)} \), every \((S, O) \in b^{(1)}\) satisfies SC rel \( f^{(1)} \). That is, \( z_1 \) is secure.

If \( z_{t-1} \) is secure, \( z_t \) is secure. \((x_t, y_t, z_t, z_{t-1}) \in W\).
Thus by (i), \( f(t) = f(t-1) \). By (ii), every \((S,0)\) in \( b(t) - b(t-1)\) satisfies \(SC_{rel} f(t)\). Since \(z_{t-1}\) is secure, every \((S,0) \in b(t-1)\) satisfies \(SC_{rel} f(t-1)\). Since \( f(t) = f(t-1) \), every \((S,0) \in b(t)\) satisfies \(SC_{rel} f(t)\). That is, \(z_t\) is secure. By induction, \(z\) is secure so that \((x,y,z)\) is a secure appearance. 

**SUMMARY**

In this chapter we have applied the mathematical model of Section II to the modeling of a secure computer system. We have defined a secure system precisely, through the definitions of security and compromise, and have given a rule of operation, \(W\), which we have shown guarantees that the system is secure in its operation.
INTRODUCTION

We attempted to provide in Section I a motivation and basis for the remainder of this paper. We pointed out three desirable properties of a model — generality, predictive ability, and appropriateness — and these were illustrated by example. Also, we discussed the general principle that the specificity of prediction is roughly proportional to the amount and level of detail of information available about the system being modeled; this was illustrated by the discussion of the spring-mass system.

Subsequently, we developed a mathematical model of general applicability to the study of secure computer systems, abstracting the elements of the model from our own and others' notions of what the real system may be like.

We then applied the model, under a given set of assumptions, to the question of security (compromise). We gave a rule by which, for the assumptions given, the system would remain secure in its operation; we also gave a proof of the last assertion.

Notice this important point: our proof did not depend on the choice of elements for the set $A$ (access attributes). This means that any set is acceptable and any access matrix is acceptable. Stated differently, we have shown that under the given assumptions security of the system is independent of the access matrix and the rules (if any) by which the access matrix is changed.
Thus, we have modeled the system in such generality that we are not in a position to investigate its viability. For, clearly, one may arbitrarily choose rules of access matrix control while retaining the property of security. Therefore, one may choose the rules in such a way as to prevent users from ever acquiring access to information; the severe danger is that a set of rules might be chosen which has an intuitive sense of correctness but which may lead the system into undesirable states.

We shall address ourselves in this section to some of the specific questions to be considered if a viable system is to be developed from our model.

**PROBLEM REFORMULATION**

One may change the system problem to be attacked in a variety of ways. In general one states a set of requirements and a set of criteria to be met. The requirements and criteria may be very general or very specific: the more specific these are, the more specific can be the behavior predicted by modeling and the greater the probability that a viable system will result from the design into which the model is transformed.

In our situation we can immediately recognize two areas of problem reformulation. First, one may change the requirements of the type we assumed in Section III. We shall, in fact, do so and derive a result from the changed assumptions. Second, one may impose criteria to be met by the access control mechanisms of the system. We shall investigate this briefly in the next two sections.
We change the assumptions we made in Section III, as shown in Table IV.

**Table IV**

**Modified Requirements**

<table>
<thead>
<tr>
<th>REQUIREMENTS</th>
<th>RAISE?</th>
<th>LOWER?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT CLEARANCE</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>OBJECT CLASSIFICATION</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>INCREASE?</td>
<td>DECREASE?</td>
<td></td>
</tr>
<tr>
<td>SUBJECT NEEDS-TO-KNOW</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>OBJECT NEEDS-TO-KNOW</td>
<td>NO</td>
<td>YES</td>
</tr>
</tbody>
</table>

**Basic Security Theorem (revised):**

Let \( W \subseteq R \times D \times V \times V \) be any relation such that

\[
(R_1, D_1, (b^*, M^*, f^*), (b, M, f)) \in W \quad \text{implies}
\]

\[
\begin{align*}
(1) \quad f_1^*(S) & \geq f_1(S) \quad \text{for each } S \in S, \\
& f_2^*(0) \leq f_2(0) \quad \text{for each } 0 \in 0, \\
& f_3^*(S) \geq f_3(S) \quad \text{for each } S \in S, \\
& f_4^*(0) \leq f_4(0) \quad \text{for each } 0 \in 0, \text{ and}
\end{align*}
\]

(ii) every \((S, 0) \in b^* - b\) satisfies SC rel \(f^*\).

Then \( \Sigma(R, D, W, z_0) \) is a secure system for any secure state \( z_0 \).

**Proof:** Let \( z_0 = (b, M, f) \) be secure.

Pick \((x, y, z) \in \Sigma(R, D, W, z)\) and write \( z_t = (b(t), M(t), f(t)) \) for each \( t \in T \).

\( z_1 \) is a secure state. \((x_1, y_1, z_1, z_0) \in W\).
By (ii), every \((S,0)\) in \(b^{(1)} - b\) satisfies SC rel \(f^{(1)}\). Since \(z\) is secure, every \((S,0)\) in \(b\) satisfies SC rel \(f\); that is, \(f_1(S) \geq f_2(0)\) and \(f_3(S) \geq f_4(0)\). By (i), we have, for each \((S,0)\) in \(b^{(1)} - (b^{(1)} - b)\),

\[
\begin{align*}
f_1^{(1)}(S) &\geq f_1^{(1)}(S) \geq f_2^{(1)}(0) \geq f_2^{(1)}(0) \\
f_3^{(1)}(S) &\geq f_3^{(1)}(S) \geq f_4^{(1)}(0) \geq f_4^{(1)}(0),
\end{align*}
\]

so that each \((S,0)\) in \(b^{(1)}\) satisfies SC rel \(f^{(1)}\). That is, \(z_1\) is secure.

If \(z_{t-1}\) is secure, then \(z_{t}\) is secure.

\((x_t, y_t, z_t, z_{t-1}) \in W\). By (ii), every \((S,0)\) in \(b^{(t)} - b^{(t-1)}\) satisfies SC rel \(f^{(t)}\). Since \(z_{t-1}\) is secure, every \((S,0)\) in \(b^{(t-1)}\) satisfies SC rel \(f^{(t-1)}\); that is,

\[
\begin{align*}
f_1^{(t-1)}(S) &\geq f_1^{(t-1)}(0) \geq f_2^{(t-1)}(0) \\
f_3^{(t-1)}(S) &\geq f_3^{(t-1)}(0) \geq f_4^{(t-1)}(0),
\end{align*}
\]

By (i), we have for each \((S,0)\) in \(b^{(t)} - (b^{(t)} - b^{(t-1)})\),

\[
\begin{align*}
f_1^{(t)}(S) &\geq f_1^{(t-1)}(S) \geq f_2^{(t-1)}(0) \geq f_2^{(t)}(0) \\
f_3^{(t)}(S) &\geq f_3^{(t-1)}(S) \geq f_4^{(t-1)}(0) \geq f_4^{(t)}(0),
\end{align*}
\]

so that each \((S,0)\) in \(b^{(t)}\) satisfies SC rel \(f^{(t)}\). That is, \(z_t\) is secure.

By induction, \(z\) is secure so that \((x,y,z)\) is a secure appearance. \((x,y,z)\) being arbitrary, \(\Sigma(R,D,W,z_0)\) is secure.
The revised theorem just proved indicates that dynamic
(i) raising of subject clearance;
(ii) lowering of object classification;
(iii) increasing of subject needs-to-know; and
(iv) decreasing of object needs-to-know
can be provided in the system without security compromise. Again,
however, the proof is independent of what is happening in the access
matrix, the subject of the next section.

We note here that our investigations into the security of a system
in the cases that a subject's clearance may be lowered dynamically,
an object's classification may be increased dynamically, and similar
changes in needs-to-know are as yet undocumented. Those investigations
lead us to believe that severe questions of the viability of the
resulting system are raised by the options listed above.

ACCESS CONTROL

In a real sense, the relation $W$ we have specified provides a
rule of access control which governs security as we have defined it.
We have also provided in the model for access control to govern
protection, privilege, and mode of use through the access matrix we
have defined.

Two problems are immediately evident. First, unless the system
guarantees the inviolability of rule $W$ our security theorem does
not apply. Second, unless we deal with some specific criteria and
rules relating to the access matrix, we can say little if anything
concerning viability of the system; again, if access matrix controls
are provided, the system must be structured so as to guarantee their
inviolability else our modeling will not apply.
Let us consider a situation in which the interaction of security control and access control can cause a compromise. Specifically, if a subject $S_i$ is allowed "append" access to an object $O_k$, a file or segment, then guaranteeing inviolability of rule $W$ means the system must prevent $S_i$ from appending information of a classification higher than that of $O_k$; otherwise we risk having $(S_i, O_k)$ in $b$, where $S_j$ has "read" access to $O_k$, while $f_1(S_j) < f_2(O_k)$ resulting in compromise. This example shows that inadequate access controls (over the "append" access of $S_i$ to $O_k$) can cause a violation of $W$ (by raising $f_2(O_k)$, contrary to our assumption up to this point), resulting in a compromise state.

DATA BASE SHARING

We have assumed a shared data base for the multi-user system but have stated no requirements nor criteria for "correct" sharing. The concluding remark of the preceding section suggests that we must do so. At least, we must specifically prevent the situation we discussed; alternatively, one might choose to change our definition of compromise. Unfortunately, a change in the definition of compromise in this situation would be in the direction of weakening rule $W$ with the result that the model will reflect the real problem less accurately than we have succeeded in doing thus far.

In addition, one may impose additional criteria relating to sharing of the data base, such as prevention of deadlock, preservation of integrity of the information, and prevention of permanent blocking—such criteria have to do with reliability of the system and therefore relate to its usefulness.
SUMMARY AND REFERENCES

In this chapter we have discussed the generalities of changing the definition of the problem to be solved. We showed an example by stating and proving the security theorem for a new set of assumptions relating to changes in classifications and needs-to-know.

We pointed out briefly that the system which one might develop from our model would have to guarantee inviolability of the rule of operation W. Techniques have been documented which use hardware, software, or combinations of these for protection of privileged algorithms; references [1,2,3,4,5,6,8,9,10] are relevant.

We discussed briefly the question of a shared data base. For a discussion of problems and a solution see [7].

In summary, we have attempted to show in this section that the model can be used to answer questions posed with a given set of requirements and criteria and to indicate that a central problem in the design of a secure system will be to certify that the access controls are inviolable.


BIBLIOGRAPHY


BIBLIOGRAPHY (Continued)


