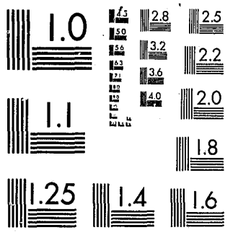
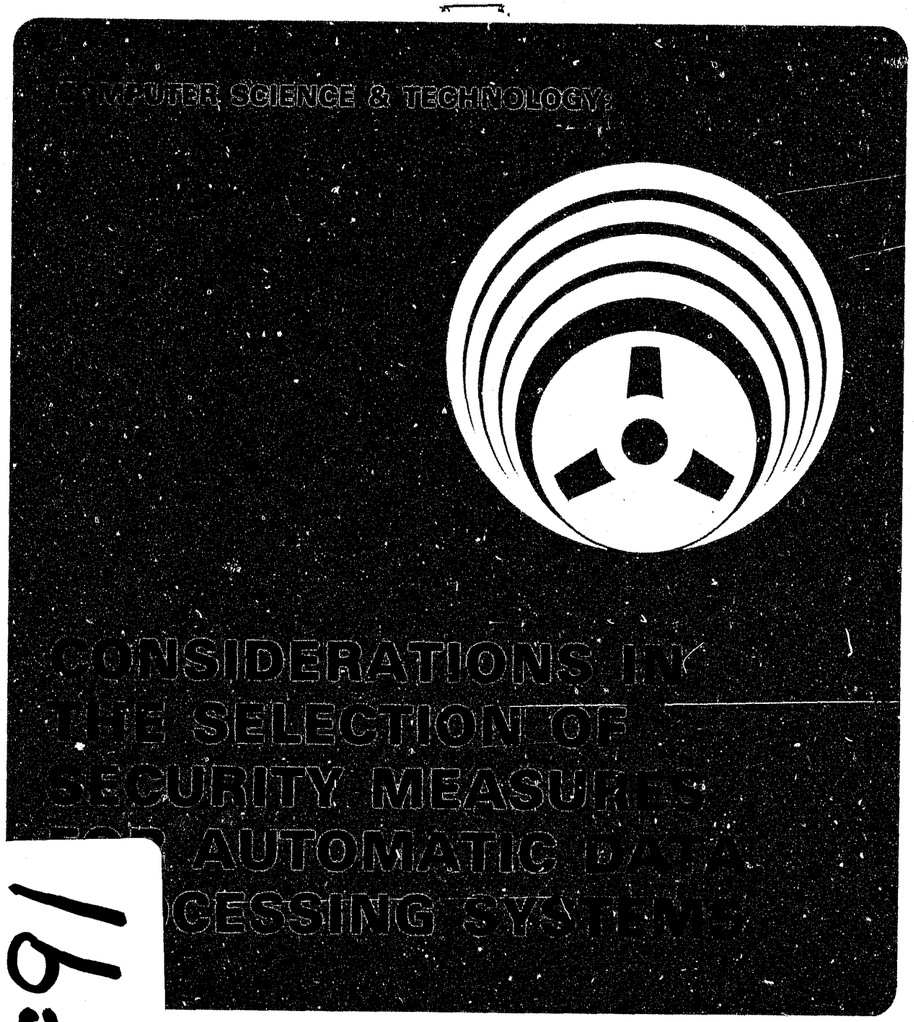


MFI



NBS Special Publication 500-33
U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards

NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards[1] was established by an act of Congress March 3, 1901. The Bureau's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the Nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, and the Institute for Computer Sciences and Technology.

THE NATIONAL MEASUREMENT LABORATORY provides the national system of physical and chemical and materials measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; conducts materials research leading to improved methods of measurement, standards, and data on the properties of materials needed by industry, commerce, educational institutions, and Government; provides advisory and research services to other Government Agencies; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

Absolute Physical Quantities[2] — Radiation Research — Thermodynamics and Molecular Science — Analytical Chemistry — Materials Science.

THE NATIONAL ENGINEERING LABORATORY provides technology and technical services to users in the public and private sectors to address national needs and to solve national problems in the public interest; conducts research in engineering and applied science in support of objectives in these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the utlimate user. The Laboratory consists of the following centers:

Applied Mathematics — Electronics and Electrical Engineering[2] — Mechanical Engineering and Process Technology[2] — Building Technology — Fire Research — Consumer Product Technology — Field Methods.

THE INSTITUTE FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides scientific and technical services to aid Federal Agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal Agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following divisions:

Systems and Software — Computer Systems Engineering — Information Technology.

[1]Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington,D.C. 20234.
[2]Some divisions within the center are located at Boulder, Colorado, 80303.

The National Bureau of Standards was reorganized, effective April 9, 1978.

COMPUTER SCIENCE & TECHNOLOGY:

Considerations in the Selection of Security Measures for Automatic Data Processing Systems

Michel J. Orceyre

IBM Corporation
10215 Fernwood Road
Bethesda, MD. 20034

Robert H. Courtney, Jr.

IBM Corporation
Neighborhood Road
Kingston, New York 12401

Edited by

Gloria R. Bolotsky

Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

Contributed to the
Federal Information Processing Standards
Task Group 15 - Computer Systems Security

U.S. DEPARTMENT OF COMMERCE, Juanita M. Kreps, Secretary

Dr. Sidney Harman, Under Secretary

Jordan J. Baruch, Assistant Secretary for Science and Technology

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, Director

Issued June 1978

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

National Bureau of Standards Special Publication 500-33
Nat. Bur. Stand. (U.S.) Spec. Publ. 500-33, 33 pages (June 1978)
CODEN: XNBSAV

Library of Congress Cataloging in Publication Data

Orceyre, Michel J.
Considerations in the selection of security measures for automatic data processing systems.

(Computer science & technology) (National Bureau of Standards special publication ; 500-33)
Supt. of Docs. no.: C13.10:500-33
1. Electronic data processing departments--Security measures. 2. Computers--Access control. I. Courtney, Robert H., joint author. II. Bolotsky, Gloria R. III. United States. National Bureau of Standards. IV. Title. V. Series. VI. Series: United States. National Bureau of Standards. Special publication ; 500-33.
QC100.U57 no. 500-33 [HF5548.2] 602'.1s 78-17838
[001.6'4]

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1978

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402
Stock No. 003-003-01946-1 Price 1.40

(Add 25 percent additional for other than U.S. mailing).

FOREWORD

The need for improved security of computer systems has risen along with the need for improved utilization of those systems. The increasing use of computers by Government and private industry for the processing, storing and communication of sensitive, as well as valuable data, has focused this need and has resulted in an intensive program at the National Bureau of Standards for improving the security that is available within a computer system. This publication is one product of this cooperative program between Government and industry.

The information in this document was submitted to the Federal Information Processing Standards Task Group 15 (Computer Systems Security) as an appendix to a risk analysis document authored by Robert H. Courtney, Jr. The information was considered valuable by the TG-15 participants as a tutorial on what to consider using for security improvements after a risk analysis has been performed. The steps of a computer security program include:

- o Perform a security risk analysis;
- o Consider all security measures available;
- o Select those measures that minimize the risk at a minimum cost;
- o Implement those measures that are feasible;
- o Evaluate their effectiveness and actual cost;
- o Restart the process.

The information in this document is intended to outline those security measures which may be selected and used in this process.

Although Task Group 15 was terminated as a formal public advisory committee, the work initiated by the group and the contributions made by its participants will be utilized in products of the NBS computer security program and will be made available for use by Federal ADP organizations and private industry.

Dennis K. Branstad
Past Chairman
FIPS Task Group 15

PREFACE

This document presents an overview of currently known methods and techniques for securing information processed by computers and transmitted via telecommunication lines. Originally contributed by the authors to the Federal Information Processing Standards Task Group 15 on Computer Systems Security, this revised document is intended as a follow up document to Automatic Data Processing Risk Assessment (NBSIR 77-1228). This publication summarizes protective measures which aid in identifying controls already in use and selecting further safeguards to offset existing risks and potential losses identified by a risk analysis.

In writing this report, the authors drew from their years of experience in data security and from unpublished papers authored by them prior to 1975. The Federal Information Processing Standards Task Group 15 is grateful to Robert H. Courtney, Jr. and Michel J. Orceyre of the IBM Corporation for their generous contribution and guidance in adapting their original material to the needs of the Federal Government.

The following members of the Institute for Computer Sciences and Technology of the National Bureau of Standards are acknowledged for their efforts in producing the final version: Dr. Dennis K. Branstad, Dr. Thomas C. Lowe, Dr. Theodore A. Linden, Dr. Jason Gait, Ms. Susan K. Reed, Dr. Stuart W. Katzke, and Mr. Paul Meissner for their helpful comments; Mrs. Karen K. Toms for her patience, diligence and sustained effort in the typing of this report.

ABSTRACT

The authors introduce the readers to presently known methods and techniques for protecting data in an ADP facility and during transmission. The material is presented as an aid in evaluating and selecting security measures following the identification of existing risks and potential losses via a risk analysis.

Key words: Auditing; authorization; computer security; cryptography; device identification; distributed processing; identification; personal identification; security; surveillance

TABLE OF CONTENTS

1. INTRODUCTION

This publication has been prepared to provide general guidance for the consideration of candidate security measures for ADP. It is anticipated that appropriate measures will be evaluated and selected only after a detailed assessment of the potential losses which these measures are to prevent (see Reed in the Bibliography). The cost justification for any measure or combination of measures must be that the problems which the measures obviate would result in a cost significantly more than that of the corresponding security measures, or that the net "cost efficiency" of the resulting system (in terms of reliability, manageability, predictability and so forth) clearly justifies the cost of security enhancements.

An array of security measures is detailed in the following sections. The intent is to familiarize readers with the protective measures that should be considered for inclusion in systems, how these features integrate into a coherent, consistent mechanism, why they are needed, and how they might be used. The following summarizes management objectives for a secure ADP system, classes of protective measures for achieving these objectives, and other concepts related to the system integrity and operational reliability of a secure system.

1.1 Management Objectives

To protect data assets adequately from accidental or unauthorized intentional disclosure, modification and destruction, installation management should select security measures that will accomplish the following:

- Enable installation management to hold each user personally accountable for his activities on the system.
- Bestow the least access capability necessary to enable users to get their work done.
- Identify and reduce the frequency and impact of errors and omissions on the part of system users.
- Ensure that it is difficult for users to defeat constraints or to misuse authorized capabilities, and to ensure that any effects of such defeat or misuse are localized and minimized.
- Impose a high actual and discernible risk of apprehension and significant penalty for users' misuse of the system.
- Give protection, not only against normally high-exposure threats, but also against normally low-exposure threats that may be highly significant in a particular environment, industry or installation.
- Yield positive contributions in terms of asset protection and increased stability, manageability, predictability, reliability, and imperturbability of the system that can be seen to outweigh any unavoidable negative impacts such as performance degradation and human inconvenience.

In other words, security measures should help the system owner to institute and enforce prudent protection, and they should be such that in the event of wrongdoing, unquestionable evidence of the nature of the activity and the identity of the wrongdoer is available.

1.2 Protective Measures

The various classes of hardware and software protective measures support:

- Authorization (definition and control) of system activities involving interactions among people, data, programs, devices and other named system resources.

- o Surveillance of system activity - means of achieving strict personal accountability of people for their actions.
- o Positive, unique identification of people, devices and other named system resources.
- o Data Encryption
- o System integrity - means of achieving hardware and software integrity, physical security and protection against wiretapping and electronic and acoustic eavesdropping.

1.3 Other Considerations

Coherence and consistency of the set of all security measures at the system level are not the only concerns faced by those who must select security measures. System integrity and operational reliability must at no time be impaired. The following topics related to system integrity and operational reliability are, therefore, discussed in this report.

- o Performance, storage requirements and human factors
- o Optionality of functions
- o Distributed processing architectures
- o Testing procedures
- o Auditing
- o Documentation

Other concerns requiring attention, but not discussed in this text, are:

- o Recoverability
- o Effects of maintenance and servicing on the protection mechanisms

2. AUTHORIZATION

Authorization is the means whereby management can control interactions among people and named system elements, including devices, software, communication lines, and data objects such as indexes, records, and fields within those records. The two steps required for providing this ability to enable or inhibit such interactions are:

- o Rules definition (authority setting)
- o Rules execution (access control)

An authorization mechanism should, to the degree needed and specified by installation management, enable

- o only authorized users to perform...
- o only those functions which they are authorized to perform...
- o only upon those data to which they are authorized access, using...
- o only those hardware and software resources which they are authorized to use.

In general, the principle of "least privilege" should govern. The less a person using the system is allowed to do (consistent with the work he is required to do), the safer will be the system's other users, and the individual's own processes and resources.

2.1 Named Elements

The authorization mechanism should generally operate upon names of elements and should be invoked when one named element refers to another (e.g., an access request, a call for execution, a system service request). The mechanism should be invoked at the point where the controlling process resolves such symbolic references.

Named elements which are candidates for authorization control include:

- o Persons
- o Devices (terminals, controllers, printers, CPUs, etc.)
- o Data objects (data sets, segments, libraries, records, etc.)
- o Executable objects (transactions, commands, programs, etc.)
- o Storage media (cartridges, magnetic tape reels, disk packs, etc.)
- o System control objects
- o Application subsystems (both software and hardware)
- o Named groups of these elements

The ability to create named groups of system elements that can share common authorization attributes is an important administrative tool. Such groups can be treated, from an authorization point of view, as elements themselves; this enables management to classify elements and thus reduce the number of individual entities with which it must deal on a frequent basis.

It should be possible to declare an element to be a member of more than one group, and to give specific elements that are members of a group additional or reduced capabilities relative to the group. This flexibility amounts to templating (or performing complex definitions automatically) and can reduce the administrative overhead significantly.

2.2 Bases for Authorization

The authorization mechanism should prevent or allow interactions among elements based, not only upon the names of participant elements, but also upon:

- o the nature of the requested interaction (i.e., create, read, alter, append data to or delete a data object)
- o the nature of the participant elements (i.e., sensitive data must be displayed/printed at only designated output devices)
- o the testable external conditions (i.e., time of day, date, storage space available to the user, other people or processes currently active)

2.3 Hierarchies of Authorization

Installation management should be able to specify the extent to which authorization for certain kinds of interactions implies that the holder is authorized for other kinds of interactions. For example, it should be installation management's, not the designer's, decision that "create" authority includes "alter" authority for a data object. The authorization mechanism should not force such hierarchies of authority upon management.

2.4 Levels of Authority

Installation management should be able to specify that certain authorities held by a user imply his ability to bestow given authorities upon other users. For example, a person's authority to alter a data object should only imply at installation management option that he can authorize others to interact in any way with that object. In general, the three authority levels:

(1) ability to interact
(2) ability to authorize interactions
(3) ability to appoint those who may authorize interactions

should be discrete, independent conditions. None should, except at installation option, imply either of the others. This is analogous to the distinctly different authorities involved in entering a bank vault, guarding the vault (deciding who may enter), and appointing guards.

2.5 The Authorization Data

The authorization mechanism is driven by a structure of authorization data or "rules." This is, in a sense, a model of the activities that management expects and considers desirable within the system. It is important that the authorization data be correct and adequately secure, because it actually controls system activity and has great potential for disruption.

It should be very simple for management to establish, modify, delete and display the authorization data. If these processes are complex or difficult or unwieldy, errors will be more likely, disruptions may be more frequent, and use of the authorization mechanism by installations will be less likely. In the past, simple and flexible entry, modification and display capabilities have been perceived by installations to enhance the manageability of the system (quite apart from any security enhancement, if good management and security are indeed separable). Where they have not been offered, management has been reluctant to use the authorization mechanism.

The authorization mechanism, to the extent feasible, should be self-protective. Erroneous, anomalous, or inconsistent authorization data entries should be detected and reported as early as possible, hopefully at the time of entry and at least at the time they are first used in normal authorization checking.

2.6 Keeping Authorization Data Current

In all likelihood, the authorization relationships within a given system will change frequently as people, data, software and hardware change with time. The burden of keeping authorization data in line with current needs could easily grow out of hand (as can the amount of authorization data) if flexible entry, updating, and display capabilities are not provided. Since there may be significant effort involved in keeping the authorization data current and correct, management must be able to delegate this work as much as possible to administrators and to users themselves in the normal installation. Another important reason for delegating authorization responsibilities is that supported departments, those whom the applications and data are serving, must be able to control access to their own resources. However, the capability of concentrating or centralizing this work is a requirement in certain environments.

2.7 Modifying Authorization Data

The authorization mechanism should ensure that management is protected against destructive or disruptive secondary effects of modifications to authorization data. When an individual with authority to access an object and to authorize others to access the object (who, in turn, may authorize still others to access the object) is about to have his authority removed, the consequences must be well understood. One consequence (depending upon design) might be that all users whose authority originates from that individual will lose their authority when his is removed. This may be acceptable in some cases, and it may be catastrophic in others. On the other hand, if the authorities originating from his authority are undisturbed (or untraceable) when his is removed, this too can be catastrophic, or at least create an administrative burden. The authorization mechanism should enable installation management to discern the ultimate effects of such removals or modifications of authority. This requires that the authorization data, including backchained or derivative authorities and all grouping relationships, be displayable in some well-formatted structure.

2.8 Authorization Bypass Capability

Since damage to the authorization mechanism or data can disable operations, some bypass mechanism or procedure must be available so that management has the option to continue operations in an unprotected mode. Authorization mechanism design should not assume that management uniformly believes that system shutdown is preferable to interrupted or degraded protection. On the other hand, any bypass mechanism is sensitive and dangerous and must be shown to be safe from unauthorized use.

2.9 Transfer to a Back-Up System

An authorization mechanism can further complicate difficult system back-up and recovery problems. If all, or a significant part, of the system's operations must be brought up on another hardware configuration, perhaps one that must share its capacity with another conceivably "hostile" workload, then the authorization structure should be such that the move to the new system is not inordinately difficult, requiring wholesale revision or piecemeal deletion of integrated authorization data. If severe back-up/recovery difficulties are introduced as a result of the authorization mechanism, then a reluctance to employ the authorization mechanism at all will inevitably evolve -- and properly so, since back-up is such an important installation requirement.

2.10 Converting to a Controlled System

It can be a traumatic experience to transform an installation from one with little or no authorization control to one that is heavily controlled. At most installations, management simply does not possess the required specific information to create the complete set of authorization data, and the information can be difficult and costly to collect. Thus, a gradual transition is indicated. Mechanisms have been proposed to aid this process. One such mechanism is to include in the authorization mechanism a capability such that initially, while all checking based on the growing body of authorization data is done normally, no authorization "failures" cause denial of the requested interaction. Instead, the interaction is allowed to proceed and a record of the authorization failure is kept for subsequent analysis. In this way, management can correct the expected high incidence of errors during the transition without disrupting normal operations. As confidence in the authorization data increases over time, the normal authorization failure processing can be used increasingly until the system has been fully converted.

3. SURVEILLANCE

The objective of the surveillance mechanism is to ensure that management can detect and react appropriately to activities that it has determined may constitute security threats. The surveillance mechanism must provide a means of achieving strict personal accountability of users for their actions on the system. In addition to such accountability processing, the surveillance mechanism may protect in real-time against damage from certain events. It may also act as a strong deterrent to the user who might otherwise abuse his privileges but who perceives, because of the surveillance capability, that the risk of detection is unacceptably high. The following summarizes the requirements for a surveillance mechanism:

- Recognition of predesignated "trigger-events"
- Evocation of predesignated reactions to specified events
- Collection of predesignated information (journalling)
- Provision for management inspection (post-processing or real-time) of surveillance data

3.1 "Trigger-Events"

In general, any event that can be designated as requiring an authorization test is a candidate surveillance stimulus. However, there should be no designed-in constraint so that only an invoked authorization test can stimulate surveillance. The two activities--authorization and surveillance--should be independent such that an event can cause either or both.

Events such as:

- LOGON
- OPEN for write

and event characteristics such as:

- o Participant people
- o Participant resources
- o Data sensitivities
- o Numeric values of data fields
- o Times of day

are candidates for designed "trigger-events."

"Trigger Event" designations should be simple for management to establish and modify. Management should be able to add, alter, and display such specifications interactively and easily.

3.2 Reactions to Specified Events

The surveillance mechanism should provide for a number of optional surveillance reactions, depending on the nature of the detected event (an authorization test failure, for example, as opposed to a success). Selectable reactions should include:

- o "normal" journalling
- o real-time alerts to management such as a warning bell and message to a designated console
- o suspension or termination of an offending process with a variety of possible messages to the user such as true messages, misleading messages, or no message
- o automatic invocation of special monitoring of an offending process if it is not suspended or terminated (such as complete journalling of associated system activities or interactive traffic)
- o management-invoked real-time display at a designated console of the full interactive traffic of an offending terminal

3.3 Event Journal

The journal is the vehicle used for collecting predesignated information when specified events are invoked. The journal records should include, but not be limited to:

- o identifiers of all involved elements (people, devices, software, data)
- o the nature of the event
- o indication of success or failure of the event
- o security data such as:
  - authorization status
  - time of day
  - date

The surveillance mechanism design must give due consideration to the problems of archiving extensive data for what may be prolonged periods, even years. The ability to easily off-load voluminous journal data, to condense it as much as possible, and to on-load easily the same data for inspection much later in time, perhaps on a different machine complex, is important. Such capabilities should be required for many security-related activities, including internal and external auditing.

The event journal itself is a major security asset. At times it is the most important one. It must be protected from all but authorized access and, to the extent possible, from destructive conditions such as power failure to a volatile store. It should be demonstrable that the journal and the journalling process are reasonably secure and cannot be subverted easily, at least without detection.

It should be noted that, under the Federal Rules of Evidence, computer output is admissible in both civil and criminal proceedings if it is determined to be:

- o a regularly kept timely record
- o of regularly conducted business activity
- o whose preparation has been deemed "trustworthy" by the court

The first two conditions are relatively easy to establish for a well-run enterprise; the third may present a problem if it cannot be shown that the records and the recordkeeping process itself are reasonably safe from accidental and unauthorized intentional interference. This showing is not only essential to the court's determination of admissibility, but it is also an important defense against attacks upon the credibility of the output.

3.4 Management Inspection

The journal post-processing function (management inspection) should offer both interactive query and report generation functions. It should be possible for management to specify that certain reports be automatically generated periodically. If management does not have this processing flexibility and power, the likelihood is that the journal will not be inspected regularly; users will come to know this and the deterrent value will be lost.

Both the interactive query and the report generation functions should support complex Boolean and arithmetic operations upon data names, numeric content and statistical data derived from the journal contents. This would enable management to analyze patterns and departures from patterns of activity.

3.5 Other Uses

The surveillance mechanism has more uses than just support of data security and it need not exist only in that frame of reference. Accounting and recovery mechanisms, load-balancing, tuning and education tools require some of these capabilities. A design may be such that one multi-purpose mechanism can accomplish all or most of these ends.

Monitoring system use can also be employed to identify changes needed to improve efficiency of the work flow in and around the system. This utilization can result indirectly in improved security of operations. If a system user is advised that management has detected a pattern of frequent errors in his conduct of certain activities and is offering help, there will be an induced awareness on the part of the user that his activities are being reviewed. In this way, surveillance can be productively employed without the necessity of justifying it on the basis of detecting or inhibiting dishonesty on the part of the users.

4. IDENTIFICATION

Positive, unique identification of all system elements (people, devices, software, data objects) is clearly a requirement if authorization and surveillance mechanisms are implemented. The identification mechanism should be such that even in distributed intelligence configurations, where more than one identification process exists, the collective effect is that management can reconstruct the individual user associated with a journalled activity or event.

Unique identification is also fundamental to the integrity of operations. An estimated 70 percent of data processing-related losses occurring today can be prevented if personal accountability and "least privilege" authorization mechanisms are installed.

4.1 Personal Identification

A personal identification process has two parts:

(1) Identification
(2) Verification

Identification occurs when the user provides his identity, the "name" by which he is known to the system. The user's "name" is unique to him and unlikely to change. This identification will be used during subsequent authorization and surveillance processing.

Verification occurs when the individual, having provided an identifier, "proves" to the system, by passing some further test of identity, that he is, in fact, the person associated with that identifier.

4.1.1 Verification Methods

The state of the art today permits verification by testing people for:

- o something they know (e.g., key-entered verifiers, sometimes called passwords)
- o something they possess (e.g., magnetic stripe cards)
- o something they are (e.g., fingerprint)

Key-entered verifiers are commonly used, inexpensive and relatively frail. Verifiers may wittingly or unwittingly be given away without noticeable effects that would alert the user, management, or auditors.

The magnetic stripe card character string for verification is available but not yet widely used. It is a little more expensive than verifiers, and is much stronger. It may include characters that cannot be entered from a keyboard.

The last of the above verification methods involves testing for a personal, unique, and stable characteristic of the person; i.e., voiceprint, fingerprint, hand geometry, or signature dynamics. These areas have been researched with some success but, to date, no sufficiently inexpensive and reliable technologies for use with keyboard-type terminals have been developed.

4.1.2 Batch Processing

Some secure method must be available for identification and verification of individuals submitting batch jobs, either locally or at remote job entry sites. This should include provision for the user ID on a control card and the verifier either on the same card or on another card that may be randomly located elsewhere in the deck. At most installations, the single-card approach is considered secure enough. Where the job entry site is attended, the attendant can visually identify the individual submitting the job and can also check the verifier on the control card against a list of correct verifiers. Where the job entry site is unattended, the user himself should protect the card containing his ID and verifier. Where the job entry site is a "mail drop," or courier pickup/delivery station, the drop should be physically protected or the procedure changed to a more secure one.

4.1.3 Protection of Identification Mechanism

System design must take into account the need to protect the identification mechanism itself. To the extent economically feasible, the tables, profiles, other data and software routines associated with the identification mechanism should be protected against unauthorized access or undetected tampering.

Methods for preventing or detecting the illegal use of IDs are summarized below:

- o Verifiers Changed Periodically - Management should have the prerogative to decide at installation time when, and by whom, verifiers should be changed. Users might be permitted to alter their verifiers at will, or be required to change their verifiers at specified intervals. Management should also have the option to require the use of verifiers that are assigned, distributed, and changed by installation management and cannot be altered by the users.

- o Print/Display Inhibition of Identifiers - Where the feature exists in hardware, the identification mechanism must support print/display inhibition of identifiers and verifiers; otherwise, for printing terminals, the mechanism should provide a backspace/overstrike field for entry of the sensitive data as a means of concealing the data.

- o Terminal Disconnect or Lockup - The identification mechanism should support terminal disconnect or lockup after an installation-specified number of unsuccessful identification sequences. By introducing this unacceptable delay factor, casual masquerade and more sophisticated attempts to gain access (e.g., use of a programmable device, masquerading as a simple keyboard, transmitting all possible verifier combinations) can be deterred.

- o LOGON Messages - The identification mechanism should provide, at installation option, a LOGON message providing a sequence number ("your nth logon") or date/time last logged on, or both, with date/time preferred if only one can be provided in design. This enables the proper user to detect a successful masquerade under his identity.

- o Reverification Sequence - Another important capability that should be considered is a reverification sequence (required re-entry of the verifier) that could be invoked at installation option. This would enable the system to determine that the proper person is still present at the terminal after some specified period of line inactivity (for example, prior to accepting terminal input after a prolonged apparent "think time" delay, or prior to transmitting output to the terminal after lengthy transaction processing). If the proper verifier is not entered upon demand, the process should be gracefully suspended.

4.2 Device Identification

Device identification enables detection of and recovery from switched-network line problems, some limited defense against intruding alien devices, control over certain interactions (a form of authorization), and enhanced surveillance activity.

Device identification can be accomplished in several ways. Dial-up terminals, certain controllers, and some CPUs today offer a "hard-wired" factory-set identifier that is transmitted automatically by the device upon command from an attached device. On non-switched multipoint networks, or with local attachment, adequate identification is provided by the address at which a device is polled or selected (although a security exposure exists where it is trivial to swap cable connections at the controller either accidentally or intentionally, but without detection).

While these methods are satisfactory for local identification of devices, they are not necessarily satisfactory for higher level authorization or surveillance functions unless each such unique identifier is mapped somehow to a system-unique name for the device. The system-unique name is operated upon by the higher level mechanisms. As an example, where a subsystem controller may itself use station IDs in communicating with its terminals, and its own ID in communicating with the central processor, and the only journalling mechanism for the system and subsystem is in the host, and the installation needs records showing which terminals received which data, this information will not exist unless the central processor journalling mechanism somehow is informed by the subsystem controller, with each message, which terminal stimulated the message. In most of these cases, the intelligent controller itself will likely contain an authorization and/or a surveillance mechanism sufficient for its own needs. The information acted upon by the authorization and surveillance mechanisms, and the collective record of transmissions provided by the surveillance mechanisms should, in no case, be incomplete or ambiguous.

Many communications devices that accept dial-up connections are today equipped with "potential disconnect," or line break sensing equipment for detecting conditions such as line noise or transients. Communications systems design should include support for this equipment, such that when the "potential disconnect" interrupt occurs the system will verify that the expected device is still present on the line. If the device is present, the session should continue without interruption, even with no indication to the user. If the expected device is not presently on the line, the system should gracefully suspend the session if possible, for future resumption by the user with no loss of work already accomplished during the interrupted session.

If the potential disconnect sensing equipment is present but the terminal involved is not equipped with device ID, the reverification sequence should instead verify that the expected user is present on the line. Note that this implies that the active communications process must retain the device and/or personal identifiers related to the active sessions for possible use as comparands during such reverification sequences.

Good operations practice, as well as hardware integrity considerations, make most desirable the individual identification of portable media such as disk packs, tape reels, cartridges, floppy disks, and so on. This capability can be used to reduce or eliminate significant sources of lost data and processing time, such as operator mismounts, incorrect volume specification, and a number of integrity flaws that are described in the integrity section.

4.3 Software and Data Object Identification

System design must preclude unauthorized and undetected substitution of objects and of object names, and must preclude unauthorized, undetected, and untraceable replication and renaming of objects.

The possibility of accidental or intentional unauthorized substitution of one object for another with the same name is tantamount to uncontrolled modification of the original object and can be very dangerous in certain situations. It also reflects a serious system design flaw; system design should be such that identically named objects cannot coexist in the system. Positive, unique identification of all named software and data objects, whether system, subsystem, or application, is an important requirement.

Also, a serious flaw is the possibility of reproducing an object under a different name such that the system "loses track," cannot associate the replica with the original, and therefore cannot give the same protection to the copy as to the original. In systems where this can occur, the only defense is a well-operated journalling procedure, which, of course, is deterrent, not preventive.

The system must automatically provide identical protection to all copies of an object (under any name) as is provided for the original. This may be accomplished through hardware or software enforced addressability structures, object-name mapping, symbol resolution mechanisms, surveillance mechanisms, or by other means, but it is a basic integrity requirement.

5. CRYPTOGRAPHY

Cryptography (crypto) is the transformation of data from a clear form into a secret form (encryption) and the reverse (decryption) using a process intended to be fully known only to the proper cooperating communicators of the data. It is used when the medium containing or conveying the data (microwave transmissions, for example) cannot itself be protected adequately. The intent of encryption is to make intercepted data useless to the interceptor by making it too difficult or too expensive for him to derive the original clear data in time to use it for his purposes.

The list of threats against which crypto may afford the least expensive practical protection is not a long one. It includes interception of radio transmissions, passive wiretapping (recording or "listening in on" transmissions), active wiretapping (deletion, modification, or destruction of messages, or insertion of false messages, by the wiretapper), accidental substitution or deliberate masquerade of one device for another, and theft of or undetected interference with data that is resident on fixed or removable media or even, in some applications, in main storage.

Protection against threats to electronic communications is called Communications Security (COMSEC). Protection against threats to data resident in storage media is called File Security (FILESEC). To date, cryptography has not been considered to be an essential ingredient in most sets of security measures and, therefore, has not been widely used. It may be reasonably anticipated that the need for cryptography will continue to evolve and so a general understanding of the considerations associated with its use is desirable.

The National Bureau of Standards has published a cryptographic algorithm as a Federal Information Processing Standard. (FIPS PUB 46, "Data Encryption Standard," contains a complete description of the algorithm). The following paragraphs provide a brief discussion of considerations in the application of cryptography.

5.1 Cryptography for Communications Security (COMSEC)

5.1.1 Session Cryptography

Session cryptography (also called end-to-end encryption) is the encryption/decryption of data transmitted between two end-user mechanisms communicating during a teleprocessing session. Session cryptography implies integration of the encryption mechanism into participating end devices, at least to the extent that the system itself can control the setting of keys and the on and off switching of encryption devices. If there are intermediary devices along the communication path, session crypto is transparent to them. The key used in a given session is maintained at most for the duration of the session (it may be changed in mid-session if the design of the session encryption process allows for this). The key is generated randomly and is assigned to the specific session at the initiation of that session. If the key is transmitted over the network, it must be safely transmitted (itself encrypted under some other key known to each of the communicating devices) to one end-user device from the other (or to both from some third device). The key must be dynamically or, under some conditions, manually set in the participating end-user devices. Data encrypted at the originating device for a given transmission is not decrypted until it arrives at the destination device. The fact that it is encrypted need not be known to intermediary devices, since only the data portion of the communication (not link control or network control portions) is encrypted.

5.1.2 Link Cryptography

Link cryptography is the encryption/decryption of data only across the medium connecting two directly communicating devices. This is the classic cryptographic structure typically used in electronic communications. It is logically independent of the system and does not necessarily imply that the encryption mechanism is integrated into the communicating devices. It can be thought of as implemented by a pair of encryption mechanisms bracketing the line between two communicating devices; each encryption mechanism in this case would be situated between the communicating device and its modem. Setting the link encryption keys and switching the encryption mechanisms on and off may be accomplished manually rather than by the system. The link encryption mechanisms however, must not encrypt line-control information unless they are physically sited outboard of the line-control logic at each end of the link. Thus, if the encryption devices are not outboard (stand-alone), there must be sufficient intelligence in the link encryption mechanism to distinguish link-control information (not to be encrypted) from message content (to be encrypted). This implies that link encryption devices for different line disciplines must themselves be different.

5.1.3 Personal-Key Cryptography

Personal-key cryptography is the encryption/decryption of data using a key associated with (and manually set into the terminal's encryption mechanism by) an individual user. The user's personal key need not be transmitted within the system in any form under any condition. The personal key capability can be used in two fundamentally different applications. The first application is one in which a user's personal key is known to and used by the system for transmissions involving that person. The second major application of personal-key cryptography is the situation where only the terminal end of the session path encrypts and decrypts data.

5.1.3.1 User's Personal Key is Known to and Used by System

In this application, once the user's identity is established (in the clear) his personal key is loaded by the system at its end and by the user at his end, and communications henceforth during that session are encrypted under that key. In this mode, the effect is similar to that of session encryption. If there are intermediary devices, they need not be aware that the data they are forwarding is encrypted. Alternatively, the personal keys may be used only to protect a session key that is generated and distributed to the ends of a path for protecting the data.

Data is encrypted/decrypted only at the terminal and at the host for that session. If the session is between two terminals, with the host acting as an intermediary message-switching or processing device, then either both people must employ the same key or the system must

employ two personal keys and perform an intermediary decipher/encipher operation as it receives/transmits data between the two terminals. In any event, in this application, encryption and decryption occur at both ends of the data path.

If the system has been designed to handle personal keys, installation management might either require that all sessions of a given user be conducted using his personal key, or elect to leave the encryption decision up to the individual on a per-session basis. If the former requirement is implemented and the user does not load his key, his input will be deciphered by the host into meaningless characters. The user is thus forced to present his key in order to do his assigned task.

5.1.3.2 The Terminal End Alone Uses the Key

In this application, the system need not contain the individual's personal key or even be aware that encryption is taking place. The data entered into, manipulated within, and withdrawn from the system by the user remains encrypted while within the system. Individuals sharing access to the data must share the key.

The data can be processed only in a limited way (using text-editing-like functions such as block insertions, replacement, moving, and deletion) because encrypted values usually cannot be handled arithmetically and logically as can the same values in the clear; an encrypted "2" and an encrypted "3" do not add to an encrypted "5," for example. This is a very powerful method for maintaining the security of data within the system and despite its constraints, has been found useful.

5.2 Cryptography for File Security (FILESEC)

FILESEC is the protection of data that is not in transmission but is resident in a storage device. The file security encryption function encrypts data which needs this protection while it is on-line, in the library, in shipment on portable media from one place to another, or even in main storage, but not in active processing.

To achieve file security, the data and the key used to encrypt that data must be encrypted and stored in some medium. The original clear data may be recovered from the medium on the original system or on another system equipped with the FILESEC function.

A convenient and secure way to preserve the identity of the key used to encrypt the data on removable media is to assign identification characters to the individual keys. The key identification can then be recorded directly and in the clear on the medium label. The key-ID/key correlation tables can be preserved in a secure manner at all locations authorized access to the protected, stored data, with each table encrypted under a locally-assigned key. Each table should contain only the keys to those files to be used at that location or facility.

5.3 Cryptographic Keys

The most sensitive element of the encryption process is the particular key used to encrypt a given object. In fact, the usual measure of strength of the cryptographic process is the difficulty of deriving this key. The keys must be unavailable to any person or process other than those charged with generating, setting, invoking and maintaining them. Key handling must be reduced to an easily manageable physical security problem. It must be possible to show that the keys are not exposed to tampering or disclosure while they are within the system, that they are subjected to exposure only outside the system and, then, only after physical force has been used to obtain them. In this context, a terminal is outside the system.

5.3.1 Security of Keys

Keys must not exist within the system in clear form except when they are actually placed in one of the registers within the crypto device. These registers contain bit patterns currently in use as keys for ongoing encryption and decryption operations. Ideally, the crypto device should be so designed that physical access to registers within it destroys their contents. Given such protection, microprobing, for example, could not succeed in disclosing any key.

Where there is a necessity for keys to exist in some form within the system, yet outside the crypto device, such as in transmission of session keys or in tables associating specific keys with keynames and with persons, devices, links, ongoing sessions and files against which those keynames are mapped, the keys themselves must be encrypted.

5.3.2 Software

The software processes that initiate key transmissions, maintain the tables of keys in storage, and control the functioning of the encryption devices should be shown to be secure against any improper use that negates the encryption (i.e., turns it off, fixes one key permanently in the encryption device, modifies the key tables) or yields any key in the clear. For example, the basic commands that control the functions of the encryption device (including unexpected and apparently illogical coding sequences) must not be able to modify the proper behavior of the encryption device or to yield any key in the clear anywhere other than in the encryption device itself. The functions of the encryption device that must be protected include on/off switching, mode setting, encrypt/decrypt data commands, encrypt/decrypt key commands and load key register commands.

5.3.3 Key Selection

Any system process used for random key generation (e.g., session and file security keys) must be shown to be secure against tampering or modeling that result in disclosure of or accurate prediction of its outputs.

Any device provided or procedure recommended for use by installation management in physically setting keys within the system (in encryption devices directly or in the system's key tables) must be shown to be secure against tampering and against any methods of interception that could yield the keys in the clear. Such a procedure might, for example, recommend offline generation and encryption of the keys and their insertion into the system in already-encrypted form, or their insertion in the clear during some period when installation management can dedicate the system (at least the host) to this operation alone. In the latter case, the clear keys must immediately be enciphered and there must be certainty that no clear-key residue remains.

5.3.4 The Lost Key Problem

Encryption keys may become lost or unknown due to hardware malfunction, software error, or human failing in the physical key-handling procedures. When, for any reason, the key required to decrypt data is lost, the data itself cannot be decrypted and is permanently lost. It is just as difficult for the properly authorized possessor to decrypt his data when the key is unknown as it is for the hostile cryptanalyst who never had the key in the first place.

Users must recognize that there is no recovery from the unknown key situation, and must recognize the various ways that keys may become lost or otherwise unavailable. They must establish measures and procedures that can be used to minimize the effects of this situation (e.g., back-up data), or to reduce or eliminate the possibility of its occurrence (back-up copies of the keys).

5.3.4.1 COMSEC and FILESEC Applications

The loss of a key is of concern in some COMSEC and all FILESEC applications where it is impossible to recover from key loss or modification by resetting with new keys and repeating the operation. In any COMSEC or FILESEC encryption application where hardware malfunction results in undetected modification of the key in storage or in the encryption device, or results in failure to load the expected key, encryption or decryption of data will proceed using an unknown key. The possibility of recovery depends on the particular application.

In FILESEC applications, hardware malfunction resulting in failure to encrypt data is a serious security exposure but does not cause data loss. Malfunction resulting in modification of the correct key can result in data loss. When the protected data is the only existing copy, verification protocols should be used. When the only copy of the random file

key is stored on the medium with the data, damage to the volume (broken tape, etc.) may cause loss of the key and the data cannot be decrypted. Management should ensure that, where the application warrants, duplicate back-up copies of the protected data are available. Management should also ensure that the file master key is protected such that its loss is extremely unlikely or impossible. Copies of that key should be maintained in several physically secure and geographically separated locations.

In both COMSEC and FILESEC, software errors and human failings (in designing and following procedures) can result in irretrievable data losses, generally in scenarios similar to those described above for hardware malfunctions.

5.3.4.2 Session Encryption Applications

The following summarizes the types of errors that may result from a hardware or other malfunction during session encryption applications:

- o Erroneous Generation of a Session Key - The generated session key is not the intended key. The effect is not noticeable and does not diminish security (except the all "0"s and all "1"s keys in DES[1]).
- o Failure to Load New Session Keys at the Proper Time - One key is used during a series of sessions when different random keys should be selected. The immediate effect is not noticeable but security is lessened. This failure, whether induced or accidental, is difficult to detect and can be very serious.
- o Modification of or Failure to Load One of the Pair of Session Keys - For that session, the keys in use at each end-user device are different. The effects resulting from this type of failure depend on whether the session is a two-way or a one-way data transfer session.
  - Two-Way Data Transfer Session - The failure, in most cases, is immediately noticeable, either by a person who sees garbled message data at his terminal or by a processor that encounters unrecoverable character errors in its input stream. In this case, abnormal session termination should occur, the original data should not be destroyed, and a new session should be initiated with the same or different hardware and new keys.
  - One-Way Data Transfer Session - If there is no processing at the receiving end, then irregular or illegal bit patterns will be undetected and the session may go to completion. If the application is designed to destroy the original data upon session completion, that data may be irretrievably lost. Such session applications should include checking protocols that verify that the transmission was successfully completed with proper keys in use before the original data is deleted. Should a failure be detected, the session should be entirely redone.

It should be pointed out that such verification is not simple. It requires superencryption (double encryption) by the receiving device under the receiving device's own active key, and then transmission to the originating device for double decryption under that device's active key. Standard verification patterns and protocols may be employed, but these introduce some security weaknesses if their existence is known to the hostile cryptanalyst.

5.3.4.3 Link Encryption Applications

In link encryption applications, hardware malfunction cannot result in simultaneous erroneous modification of the pair of link keys (unless induced by human intervention, a problem addressed by the physical security measures protecting the linked devices) because the keys are set separately since there is no common system source.

[1] See "Guidelines for Implementing and Using the Data Encryption Standard," soon to be published by the National Bureau of Standards.

Hardware malfunction could result in failure to employ encryption at one end of the link, or in modification of one of the pair of link encryption keys. Both these errors should be detected at once because of the garbled data received at each end of the link.

In the one-way data transfer application, similar to session encryption described above, the data could be irretrievably lost. Also, because the transfer is not terminated, there could be a serious security exposure if the data were transmitted in the clear. Where malfunction results in failure to employ encryption at either end of the link (two-way transfer) the situation is very serious from the security point of view but data will not be lost.

It should be noted that where link-control information is encrypted/decrypted, discovery of malfunctions (except those resulting in no encryption at either end) should be immediate in every case. Where link encryption failure results in apparently successful transmission of data but, in fact, the data is lost, the situation differs slightly from the session application. Where link encryption is in use, the verification process may have to occur further along in the network than within the device to which the originator is immediately linked. The original and only copy of the data may have to be retained, not merely until the transmission across the first link is completed, but until the data reaches its ultimate destination; and this may take some time depending on the nature of the network and of the software application in use.

5.3.4.4 Personal-Key Applications

In personal-key applications where both the terminal and the host employ encryption, the exposures resulting from hardware malfunction and the concomitant verification requirements are as described above under session and link protection.

In personal-key applications where only the terminal encrypts/decrypts data, the data may be lost in some situations. If a malfunction results in failure to employ encryption, input data will be in the clear. This is a serious security exposure but will not result in loss of the data (it will be discovered later to be in the clear in the system). If malfunction results in use of the wrong key, there may be no recovery unless the originator has a copy of the data he entered (which should be a standard procedure for this application). If the magnetic stripe card containing the personal key has been damaged, resulting in use of the wrong key, then the data in the system (some encrypted under the correct key and some under the wrong key) should be recoverable if the original correct key is known (which should be standard procedure; management should have a record) and/or the current incorrect key is still on the stripe.

6. SYSTEM INTEGRITY

System integrity--an essential goal toward system security--is the condition of correct and predictable functioning of the total data processing operation, including hardware/software, physical security measures and operating procedures in force at the installation. System integrity also assumes that data integrity is maintained at an installation under abnormal operating conditions (e.g., malfunctions, crashes, maintenance and servicing situations) as well as normal conditions.

Hardware integrity and operating system integrity are included in the following discussion, not as a guide to purchasing systems, but to alert the security implementor to measures that may already be present in his system. Physical security and operating procedures are not discussed in this paper. See the bibliography for appropriate references.

6.1 Hardware Integrity

Hardware features that help achieve system integrity are exemplified by:

- o Error detection and correction capabilities -- such that no single element failure can result in an undetected error.

- o Power failure protection -- such that installation management can ensure that no failure will result in irretrievable data loss.

- o Positive, unique device identification -- devices attached through the switched telephone network which offer the "hard-wired" self-identification capability or the equivalent. Other devices may be identified through cabling addresses, "station ID" addressing protocols, and so on. (See Section 4.2 for a detailed discussion.)

- o Devices which offer positive verification of mechanical operations (e.g., seek verification in disk devices).

- o A print/display inhibit capability for interactive terminals -- automatically controlled by the system.

- o Devices to clear the residual contents of buffers, electronic storage areas, and all, or portions, of portable I/O media.

- o Processing units which offer read and write protection and two or more privilege states.

- o External storage devices designed so that there is no possibility of an "undetected mount" situation.

- o External storage devices which offer key-operated locks that prevent unauthorized removal of portable media.

- o A line-break sensing capability for all communications equipment. All conditions of potential disconnect/reconnect (such as transient noise or other switched-network disturbances) should be made known to the system so that the system will then be able to invoke device-ID reverification procedures.

- o A key-operated power on/off switch for remotely-located devices. Certain devices (particularly intelligent terminals and communicating typewriter devices) may have major functions (such as transmit, receive, typewriter only) controlled independently by key-operated switches or a single key-operated multi-function switch.

- o Microcode modification in any device may be controllable through a key-operated switch.

6.2 Software Integrity

Software integrity has received considerable attention in the last few years. The number of environments in which it is considered important to ensure that independent (and occasionally assumed to be mutually hostile) processes are well isolated, has grown suddenly in the last decade to include not just a handful of national defense installations, but service bureaus, educational institutions, law enforcement agencies, banks, research organizations, and many commercial enterprises.

With this increased attention, numerous research efforts are under way exploring such areas as:

- o Formal proofs of program correctness.
- o Operating system "kernel" structures that, partially because of their limited size, can be proven to isolate and control all processes and resources.
- o Automated integrity-flaw pattern recognition techniques for operating system analysis.
- o New processor architectures employing a variety of isolation and modularization approaches such as a large number of privilege states and storage access keys.

It is difficult to over-emphasize the importance to data security of some of the more recent developments in the management of programming staffs. The adoption of what is known as "structured programming," "top-down programming," or the "chief programmer" method not only promises enhanced productivity and fewer errors but also tends to force programmers into collusion with others if they are to get dishonestly conceived and written code into operation.

Except for certain transaction-driven systems in which terminal users can be effectively denied direct access to system resources, virtually all general purpose systems are to some degree dependent upon operating system integrity for the security of the data in the systems.

Operating system integrity may be described as the ability of an operating system to resist any compromise of specified or implicit security controls that may occur through misuse or manipulation of defined or undefined software interfaces.

Historically, operating systems have been designed with the assumption that user programs will be written without intent to overreach implied limits of isolation. Designers apparently believed that no program, for example, would supply an unexpected parameter value, or improperly attempt to gain supervisor state. Also, historically, a great deal of money has been spent by vendors and their customers in modifying code to be in line with more realistic design assumptions. With more recent systems, fewer assumptions about the benevolent behavior of programs have been made.

Operating system integrity does more than enhance data security. One important and very desirable effect is a significant reduction in system incidents. A high-integrity operating system is one in which all significant interfaces must be formalized and protected. Such an operating system, well-insulated against damaging and destructive effects of erroneous code within itself and its application programs, will be more stable and thus will enable more predictable, trustworthy operation of the entire data processing resource.

7. PERFORMANCE, STORAGE REQUIREMENTS AND HUMAN FACTORS

It is widely assumed that security features, functions and procedures are invariably costly to an installation in terms of performance degradation, storage requirements, human resentment and enforced awkwardness. Experience shows that while some of this is true (and certainly it can be made to be true), it is largely fallacious. Installations that have taken the trouble and spent the money to achieve high levels of security and to analyze the results of this activity have often found significant benefits that more than justify the security effort.

Among the positive side effects which installations have unexpectedly encountered are improved total performance of the system (higher reliability and availability because of its increased predictability) and of the entire installation (overall operation and threats to the smooth continuity of that operation are better understood and can be better controlled). Once users and management recover from the shock of change and fully understand the improved protection and service achieved, human acceptance problems disappear rapidly.

Nevertheless, product designers have the responsibility of minimizing impacts in each of these areas. Generally, good design practices should yield good performance, storage utilization and human factors. The following guidelines highlight significant objectives:

- o Performance degradation, if any, should be directly proportional to the extent to which management employs available security features and functions.
- o There should be no measurable performance degradation associated with the use of identification/verification mechanisms.
- o Degradation should be expected with use of access control mechanisms, but only as a function of the degree to which the full capability of the mechanism is employed for a given operation. A reasonable objective should be not more than 5 percent degradation (job running, response time, and system-wide measurements) attributable to any use of an authorization mechanism, however complex.

- o Degradation due to real-time surveillance activity (including journalling but not post-processing or data reduction) should be a function of the degree of use of the mechanism.
- o Degradation attributable to integrity mechanisms should be imperceptible.
- o To the extent possible, degradation should be experienced only by a process invoking a given security mechanism.
- o To the extent possible, security-related software design should not only minimize performance impact, but must take pains to make efficient use of storage, especially real main storage.
- o Security functions should be designed with great care, keeping management's objectives of strong protection, good cost/performance and high usability in mind. The user should be able to perform with ease all actions required of him. He should be unaware of and unhampered by the added protection. Security features and functions should be sufficiently flexible that managements in widely varying environments can adapt them to local security needs while in no case being forced to impose an unrealistically burdensome working environment upon the individual users.

8. OPTIONALITY FUNCTIONS

The ability to include, to exclude, and to define parameters for certain security functions is an important consideration where performance and usability are concerns. Security features and functions should be optional so that users who do not need them suffer the least possible cost, performance, or storage utilization penalty.

Optional functions should be available both to the installation management and to the individual user. Management must be able to specify which functions shall be included in all operations; for example, which personal identification and verification procedures must be followed, which events will always be journalled. Management should also be able to permit individual users to make certain security specifications regarding their own operations, such as electing to have individual sessions encrypted, or to have LOGON personal ID verification or additional journalling for specific events when management does not generally so require.

Security functions should be designed so that installation management (or the users themselves, where management so specifies) can establish default, or "automatic" authorization, surveillance, or other security-related attributes for resources and operations under their control. The accessibility of optional security functions must, in no case, enable users to reduce security by overriding management-specified procedures.

9. DISTRIBUTED PROCESSING

The statements in this paper are worded as though a system configuration includes only one control program. Obviously, such is not always the case. Increasingly, configurations include more than one control program in the form of networks and hardware subsystems. It is not our intent to recommend or imply that identification, authorization, surveillance, or integrity mechanisms must exist physically in one place in a given system.

The intent of distributed processing is to make overall system operation more efficient in some way. No security mechanism should lessen this efficiency by requiring that some function (e.g., authorization) be kept inboard or in the master control program when it could or should be placed partially in an outboard processor or internal software subsystem which controls a subset of system resources.

It is important that security mechanisms that are distributed throughout the entire system be as effective in enabling management to protect assets as though they were centralized. If a number of different processes throughout the system keep activity journals, those journals collectively should yield an accurate, usable record of the activities that management wants journalled, and should enable management to easily reconstruct the entire flow of events that was initiated by a given user, regardless of how that user was connected to the system.

Where subsystems strive to be self-contained and self-supervisory, as do certain industry-oriented subsystems, appropriate security mechanisms must be placed with each. Their design must be such that management can exercise control over user activities, derive activity records, and maintain personal accountability of users for their actions.

Often, processes are distributed to permit continued operations when a portion of the system, for any reason, is not functioning. For example, a system may normally have the host down on weekends, or may lose communications with the host because of some disruption of the lines or the host operation. In all cases, security capabilities under restricted operational conditions must be commensurate with the limited functions remaining.

10. TESTING PROCEDURES

Testing of security mechanisms can be difficult. It is required not only that they do all that they are supposed to do, and do it correctly, but also that they do, or allow, nothing that they are not supposed to do.

Since testing a negative proposition is difficult, the specifications and design of the security mechanisms must be clear, complete, and if possible, all in one place so that conducting adequate reviews and constructing adequate tests of the mechanisms is simple and can be carried on throughout the development process. If this is not the case, the probability of design oversights and flaws will be high.

11. AUDITING

It is probably fair to say that in the course of the switch-over from manual systems to automated electronic systems in the past three decades many well established and well understood auditing tools and practices have been neglected. Manual systems were developed over a very long time, were carefully studied by the auditing community, and many "classic" auditing safeguards were developed and widely used in those systems. In the rush to automate, these safeguards were neglected in favor of increases in "useful function." Costs of application development have been high, and relatively little has been spent on adapting the manual auditing practices and techniques to the ADP environment.

The management and auditing communities have recognized over the past few years that enterprises are exposed to losses because ADP systems are insufficiently auditable. Today, the situation is seen as sufficiently alarming that a great deal of resource is being applied to try to rectify it.

The lack of opportunity for independent, detailed examination of the computerized system is a principal problem of auditing today. Most audits must be conducted around the computer, because they cannot go through it. Insufficient information is captured and surfaced and controls needed to interpose testing do not exist. Some examinations are achieved through informal modes of operation tolerated by the existing controls, but these modes are not recognized formally and could be eliminated in a tightening of controls. Restoring the level of auditability that was available with manual systems, and supporting--even augmenting--it by formal ADP-oriented audit functions, are the principal goals of ADP auditors today.

System, subsystem, and application designers must attempt to understand the needs of the auditors and include functions within their designs that will improve the auditability of installed systems.

The auditor is in a special position with regard to access to the system resources and control functions. He is a potential security threat, because his work requires broad access capabilities. Both the security functions and the auditing functions must be designed to ensure that the auditor's operations are properly controlled and that he can be held properly accountable for his activities.

11.1 Auditing Techniques

Auditors examine productive processes and control processes and draw inferences about the results of each. They also examine results and draw inferences about the processes that produce them. In addition, they examine control data retained by the computing system for all processes. Examinations of these areas are uneven and variable over time. They stress what is new, sensitive, representative, suspect, or otherwise worthy of special attention.

Auditor's examination techniques may be either static or dynamic. A static examination inspects a "snapshot" of the system; it determines the character of processes or results at some point in time. A dynamic examination inspects processes in operation, and looks at results as they are formed. Any examination may inspect either real activities or activities performed solely to exercise the examination procedure.

11.2 Static Examination

Static examination of productive processes requires that the auditor create flow diagrams of programs, test plans for programs under development, and make comparisons of successive versions of programs to verify that changes are authorized. The auditor needs automated tools for these activities. Such tools include program logic analyzers that describe the program's control flow and functions, mapping mechanisms that show how accurately different specification materials (objectives, functional specifications, logic diagrams, code, etc.) relate to each other, program test case generators, program code comparison routines, and so on.

Static examination of the results (data outputs) of productive processes requires that the auditor inspect all or samples of the data against explicit criteria and manipulate, summarize, and generate reports from the data. General-purpose data processing functions are usually sufficient for such examination.

Static examination of control processes is not feasible.

Static examination of the results (journals or logs of activity) of control processes does not differ significantly from static examination of the results of productive processes.

Static examination of control data (e.g., data management format indicators, authorization tables, etc.) requires that the auditor have ready access to this information. He needs authority within the system framework to display all such information, and should be able to do so simply, with well-formatted outputs.

11.3 Dynamic Examination

Dynamic examination looks at a live system during the processing of real data. The examination is not continuous but does involve diversion of control from the real processing to the audit activity; the real processing is suspended but should not be otherwise affected, except that it might be terminated if something unusual is discovered.

The dynamic examination is typically triggered by one of a number of pre-specified events (events selected by the auditor) and control is then diverted to the auditor's routine. The kinds of processes (responses) initiated by the triggers are established by the auditor in advance.

The trigger/response relation is called the link and may be fixed, variable, or conditional. The fixed link is a simple, permanent relation of trigger and response, and cannot be changed. It may be built in or "hard-wired." The variable link is also simple but it can be changed. For all executions, the response is the same until the auditor, externally, changes the response by defining a new one for that trigger. The conditional link is a set of responses, any of which may be selected in a given instance on the basis of trigger-event characteristics and the current state of the system. The set of responses and the definition of conditions determining their selection (i.e., decision rules) are provided by the auditor.

To conduct dynamic examinations, auditors must employ tools that are fixed, variable, or conditional links. Of these, the fixed link is the least useful since it is local and inflexible and will only detect a limited set of tampering cases. The variable link, because it is simple, offers better performance than the conditional link and is preferable where a simple link is sufficient. Because the variable link is limited (i.e., one trigger, one predetermined response action) the conditional link is generally preferable. It is most flexible and can handle a large number of conditions present at the trigger event requiring a correspondingly large variety of responses. The responses may be selected either directly, according to sensed input conditions, or indirectly, according to further computation within the response mechanism.

Among the dynamic examination auditing tools required, in addition to several data-retrieval and data-manipulation tools used in static examination, are those which support tag-trace operations, parallel operation, test monitoring, and input control.

Tag-trace is an auditing operation in which a tag, or special data field not accessible by the general user, indicates to the auditing process that the tagged record is to receive special processing. The recognition of the tag, the special processing, and the journalling of the tagged item's activity together comprise tracing.

Parallel operation is the interleaved execution, upon test data, of real "production" code and of test code (designed to accomplish the same processing and outputs). Interleaving may be at the machine instruction level or at some higher level (subroutines, etc.) if more convenient. The intent is to determine the integrity of the production software, the accuracy of its operation, and any evidence of tampering.

Test monitor functions enable the auditor to generate input streams for processes, to record the execution of control paths (revealing unexpected code and untested paths), and to assess the validity of outputs resulting from known controlled inputs.

Input control operations seek to ensure that the system is properly accepting correct inputs, properly rejecting incorrect inputs, and properly accounting for and reprocessing the rejected inputs when they have been corrected.

It should be noted that many of the functions described under Surveillance are useful for auditing operations.

12. DOCUMENTATION

It is not a trivial task to design security features nor to implement them so that they are properly integrated into systems. Neither is it simple to understand and employ them correctly.

Proper planning, design and implementation of security features cannot be accomplished unless the requirements and proposed features are addressed explicitly and in detail in formal documentation at every stage of development. They require separate treatment in such documentation. If discussions of security measures are scattered throughout the documentation, the material may not be coherent or consistent.

BIBLIOGRAPHY

[1] "42 Suggestions for Improving Security in Data Processing Operations," IBM Publication G520-2797

[2] Abbott, R.P.; Chin, J.S.; Donnelley, F.E.; Konigsford, W.L.; Tokubo, S.; Webb, D.A.; Linden, T.A. (editor), "Security Analysis and Enhancements of Computer Operating Systems," NBSIR 76-1041, April 1976

[3] "An Executive's Guide to Data Security," IBM Publication G320-5647

[4] Berg, John L., "Exploring Privacy and Data Security Costs - A Summary of a Workshop," NBS Technical Note 876, August 1975

[5] Branstad, Dennis K., "Encryption Protection in Computer Data Communications Systems, Fourth Data Communications Symposium," Quebec, Canada, October 7-9, 1975

[6] Branstad, Dennis K. (editor), "Computer Security and the Data Encryption Standard," NBS Special Publication 500-27, February 1978

[7] Cole, Gerald D., "Design Alternatives for Computer Network Security," Vol. 1, and Heinrich, Frank, "The Network Security Center: A System Level Approach to Computer Network Security," Vol. 2, NBS Special Publication 500-21, January 1978

[8] "Considerations of Data Security in a Computer Environment," IBM Publication G520-2169.

[9] "Data Security and Data Processing," Volumes 1-7, Joint Study by IBM Corporation, Massachusetts Institute of Technology, TRW Systems, Inc., and the Management Information Division of the State of Illinois, IBM Publications G320-1370 through G320-1376

[10] Feistel, H., "Cryptography and Computer Privacy," Scientific American, Volume 228, Number 5, May 1973

[11] Kent, Stephen T., "Encryption - Based Protection Protocols for Interactive User - Computer Communications," Technical Report 162, Laboratory for Computer Science, Massachusetts Institute of Technology, May 1976

[12] Linden, Theodore A., "Operating System Structures to Support Security and Reliable Software," NBS Technical Note 919, August 1976

[13] Martin, J., "Security, Accuracy, and Privacy in Computer Systems," Prentice-Hall, Englewood Cliffs, New Jersey, 1973

[14] NBS FIPS Publication 31, "Guidelines for Automatic Data Processing Physical Security and Risk Management," June 1974

[15] NBS FIPS Publication 39, "Glossary for Computer Systems Security," February 15, 1976

[16] NBS FIPS Publication 41, "Computer Security Guidelines for Implementing the Privacy Act of 1974," May 30, 1975

[17] NBS FIPS Publication 46, "Data Encryption Standard," January 15, 1977

[18] NBS FIPS Publication 48, "Guidelines on Evaluation of Techniques for Automated Personal Identification," April 1977

[19] Orceyre, M.J., "Data Security," Journal of Chemical Information and Computer Sciences, Volume 15, Number 1, February 1975

[20] Pomeranz, F., "Securing the Computer," Coopers and Lybrand, New York, 1973

[21] Proceedings of the IBM Data Security Forum, IBM Publication G520-2965, September 1974

[22] Proceedings of the IBM Data Security Symposium, IBM Publication G520-2838

[23] "Privacy Act Implementation Guidelines and Responsibilities," Office of Management and Budget, Circular Number A-108, Federal Register, Volume 40, Number 132, Page 28947, July 9, 1975

[24] Privacy Act of 1974, Public Law 93-579, December 31, 1974

[25] Reed, Susan K., "Automatic Data Processing Risk Assessment," NBSIR 77-1228, March 1977

[26] Renninger, Clark R. (editor), "Approaches to Privacy and Security in Computer Systems," NBS Special Publication 404, September 1974

[27] Renninger, Clark R. and Branstad, Dennis K. (editors), "Government Looks at Privacy and Security in Computer Systems," NBS Technical Note 809, February 1974

[28] Ruthberg, Zella G. and McKenzie, Robert G. (editors), "Audit and Evaluation of Computer Security," NBS Special Publication 500-19, October 1977

[29] Sykes, David J., "Protecting Data by Encryption," Datamation, August 1976

[30] Wood, Helen M., "The Use of Passwords for Controlled Access to Computer Resources," NBS Special Publication 500-9, May 1977

NBS-114A (REV. 11-77)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBS SP 500-33	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE
COMPUTER SCIENCE & TECHNOLOGY:
CONSIDERATIONS IN THE SELECTION OF SECURITY MEASURES FOR AUTOMATIC DATA PROCESSING SYSTEMS

5. Publication Date: June 1978

6. Performing Organization Code: 640.01

7. AUTHOR(S): Michel J. Orceyre, Robert H. Courtney, Jr. / Edited by Gloria R. Bolotsky

8. Performing Organ. Report No.

9. PERFORMING ORGANIZATION NAME AND ADDRESS
IBM Corporation / National Bureau of Standards, Department of Commerce, Washington, D.C. 20234

10. Project/Task/Work Unit No.: 6401112

11. Contract/Grant No.

12. Sponsoring Organization Name and Complete Address *(Street, City, State, ZIP)*

13. Type of Report & Period Covered: Final

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

16. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)*

The authors introduce the readers to presently known methods and techniques for protecting data in an ADP facility and during transmission. The material is presented as an aid in evaluating and selecting security measures following the identification of existing risks and potential losses via a risk analysis.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*
Auditing; authorization; computer security; cryptography; device identification; distributed processing; identification; personal identification; security; surveillance

18. AVAILABILITY
- [X] Unlimited
- [ ] For Official Distribution. Do Not Release to NTIS
- [x] Order From Sup. of Doc., U.S. Government Printing Office Washington, D.C. 20402, SD Stock No. SN003-003
- [ ] Order From National Technical Information Service (NTIS) Springfield, Virginia 22151

19. SECURITY CLASS (THIS REPORT)	21. NO. OF PAGES
UNCLASSIFIED	33
20. SECURITY CLASS (THIS PAGE)	**22. Price**
UNCLASSIFIED	$1.40

USCOMM-DC 66035-P78

☆ U.S. GOVERNMENT PRINTING OFFICE: 1978—261-238/149

END