Division of Research
Graduate School of Business Administration
The University of Michigan

June 1979

IMPACT OF DISTRIBUTED COMPUTER SYSTEMS ON AUDITING

Working Paper No. 182

David G. Carney
U. S. Department of Navy

Alan G. Merten
The University of Michigan

Impact of Distributed Computer Systems on Auditing

David G. Carney and Alan G. Merten

One of the major impacts the advent of computers had on business was the centralization of data and processing activities. Data that were previously stored in a filing cabinet in an office are now stored in a computer in another room, building, or city. Similarly, processing is now performed at a central location. More recently, however, hardware and system advances have led to the concept of distributed computer systems. In a distributed computer system, local data would be stored and processing performed on a computer in or very near the office. This local computer would in turn be connected to computers in other offices, thereby permitting the sharing of data and processing.

At a macro level, distributed systems are conceptually appealing to the business community. At a micro level, distributed systems offer a significant technological challenge to the designers and implementers of such systems. The purpose of this article is to discuss four common definitions of distributed systems and audit problems encountered with distributed systems, and to propose some audit techniques to resolve those problems.

1. BACKGROUND

Within the last three years there has been increasing debate in the computer and data processing literature as to what constitutes a distributed system, how it should be controlled, and how data assets should be safeguarded. Although relatively few distributed systems have become commercially operational, researchers are currently identifying technological and organizational problems which have

hindered implementation. If historical patterns of development in other areas of computer technology prevail in the case of distributed technology, a significant number of distributed systems will soon be operational.

A myriad of definitions exist for distributed systems. There are, however, four basic definitions which embrace the major conceptual classifications. Each presents a different set of audit problems.

Distributed Processing

> Distributed processing is a technique that provides maximum utilization of computer power by sharing the processing load among many computers. Data editing, error checking, and arithmetic functions may be accomplished by the remote computers as well as by the computers at a central site.

Figure 1 is a schematic representation of this definition. In an attempt to capitalize on a "new concept" and to appear at the technological horizon, this definition or similar ones are the type often identified with vendor product brochures. It is this definition that has led many to the impression that distributed systems manifest no significant threat to auditability through existing EDP audit procedures.

According to the above definition, distributed processing is essentially a centralized system which provides some input verification, input-output reformatting, and perhaps data concentration on remote computers. Processing focus continues to be on a large, centralized main frame. Terminals which contain small computers (called "intelligent terminals") have been used for several years to perform editing functions and validation routines for second generation on-line information systems.

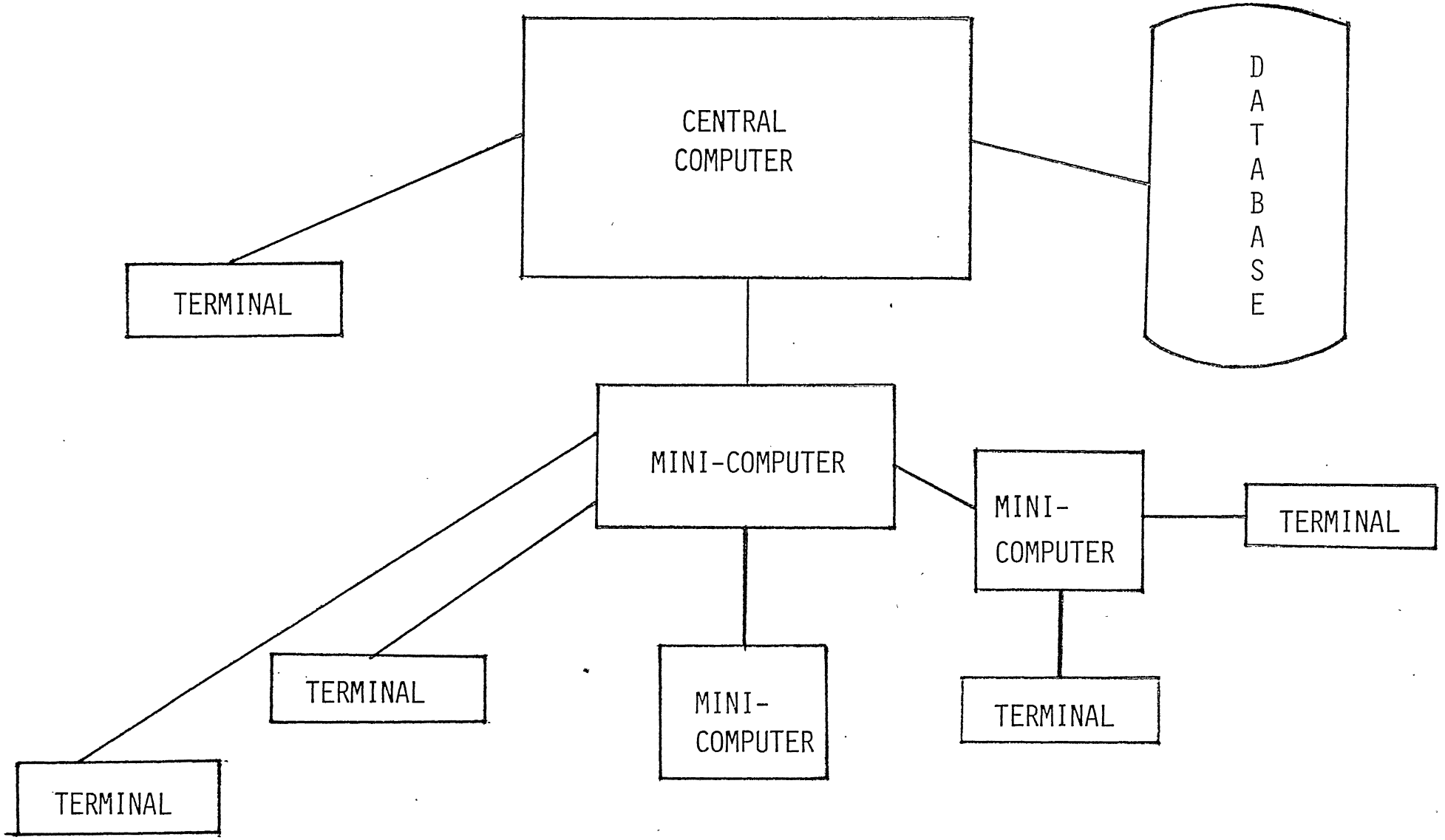


FIGURE 1 DISTRIBUTED PROCESSING

The most significant impact of systems of this type is the increased impetus for movement of data entry responsibility back to the originator (user). Interactive usage necessitates that the system assume greater responsibility for emulating the organizational division of duties. Ability to reformat data input from a single terminal entry makes it possible to initiate several applications without further input processing. Therefore, a review of systems of this type must fully analyze the system's capabilities to ensure awareness of all data file modifications which may result from a single input.

Distributed Data Systems

> Distributed Data Systems are generally those applications that are relatively large and widespread and that have partitioned and disbursed processing and data over a geographic area (Carney 1976). Figure 2 schematically represents this definition.

Distributed Data Systems (DDS) begin to provide the flavor of distribution, focusing primarily on geographic dispersion of the processing and storage of data. Skeleton files (containing detailed or summary information from the master data base) support processing that can occur on the remote computers.

Technological problems of auditability encountered in DDS are somewhat similar to those encountered in higher-level second generation systems. Although on-line interaction exists, it is often confined to a user mini-computer, providing neither on-line access to the host master computer nor to the master data base. As with the "distributed processing" definition, processing focus continues to be on a large centralized computer.

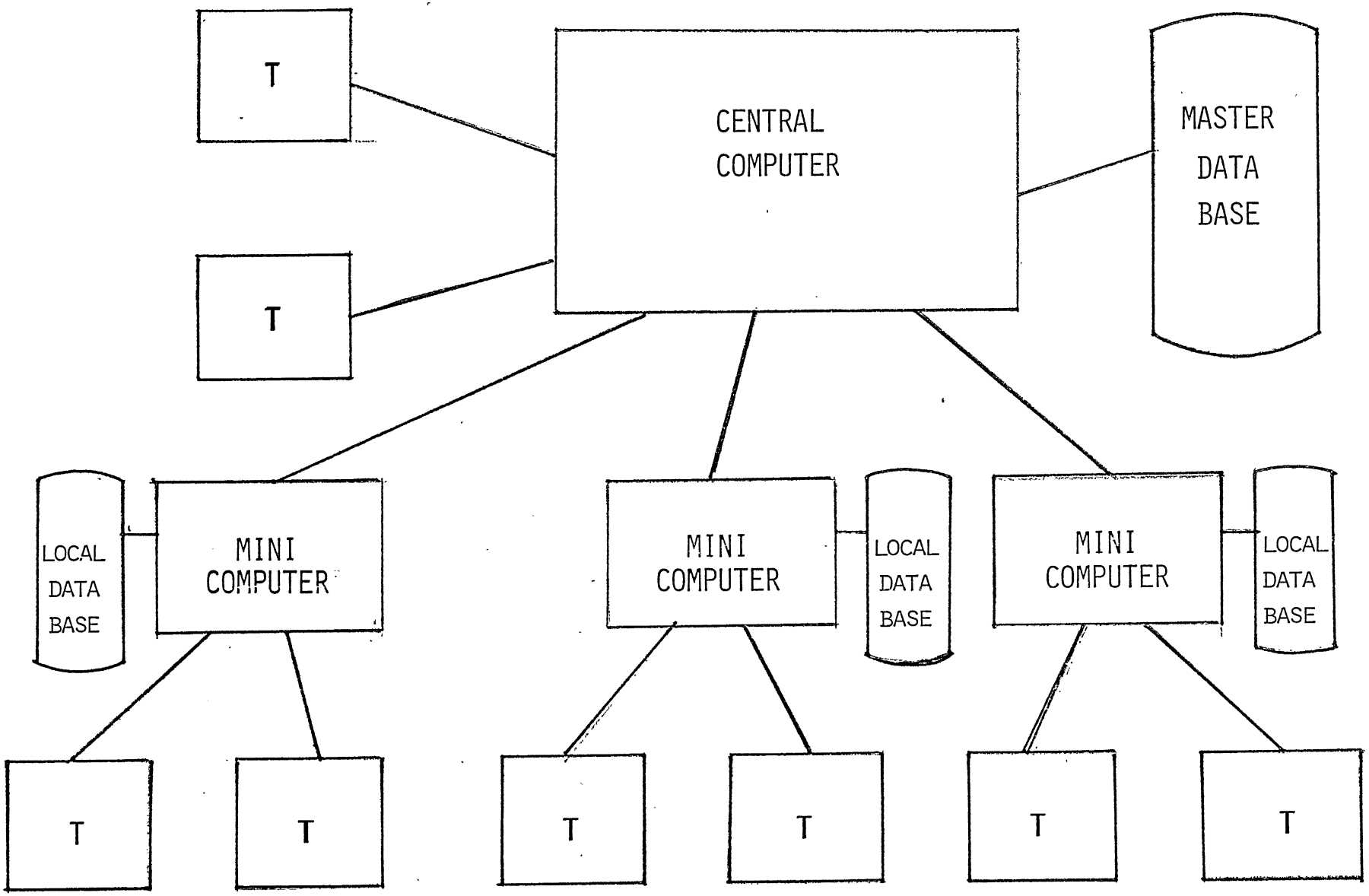


FIGURE 2 DISTRIBUTED DATA SYSTEMS

The primary advantage offered by a DDS is that remote site users enjoy high performance support for data most frequently required at that site without competing among the universe of users. System response time should be very good. Skeleton databases contain data in which local updates are reflected in support of user inquiries or other processing activities. Updates to the same data initiated at other remote sites, however, are not reflected until a batch update occurs, re-initializing all or selected skeleton files and, possibly the central database. Transactions against non-local files are processed in a manner similar to the "distributed processing" concept in which the nodes serve as a preprocessor of the transaction.

As the time interval between reconciliation of the databases at the central and remote computers decreases, users' perception of the data as live increases. Concomitant with decreased inter-update times is an inherently short audit trail life. The life of the audit trail is controlled primarily by the specific computer and communications technology employed among the various components of the DOS.

Partially Distributed Systems

Figure 3 indicates an increased level of system complexity. Direct node-to-node communication has been introduced through network processors which are communications computers.

Each node configuration operates under separate operating systems. The interdependence of processors (computers) is based upon the degree to which file directories are distributed, the method of file splitting, and user demands for non-local data.

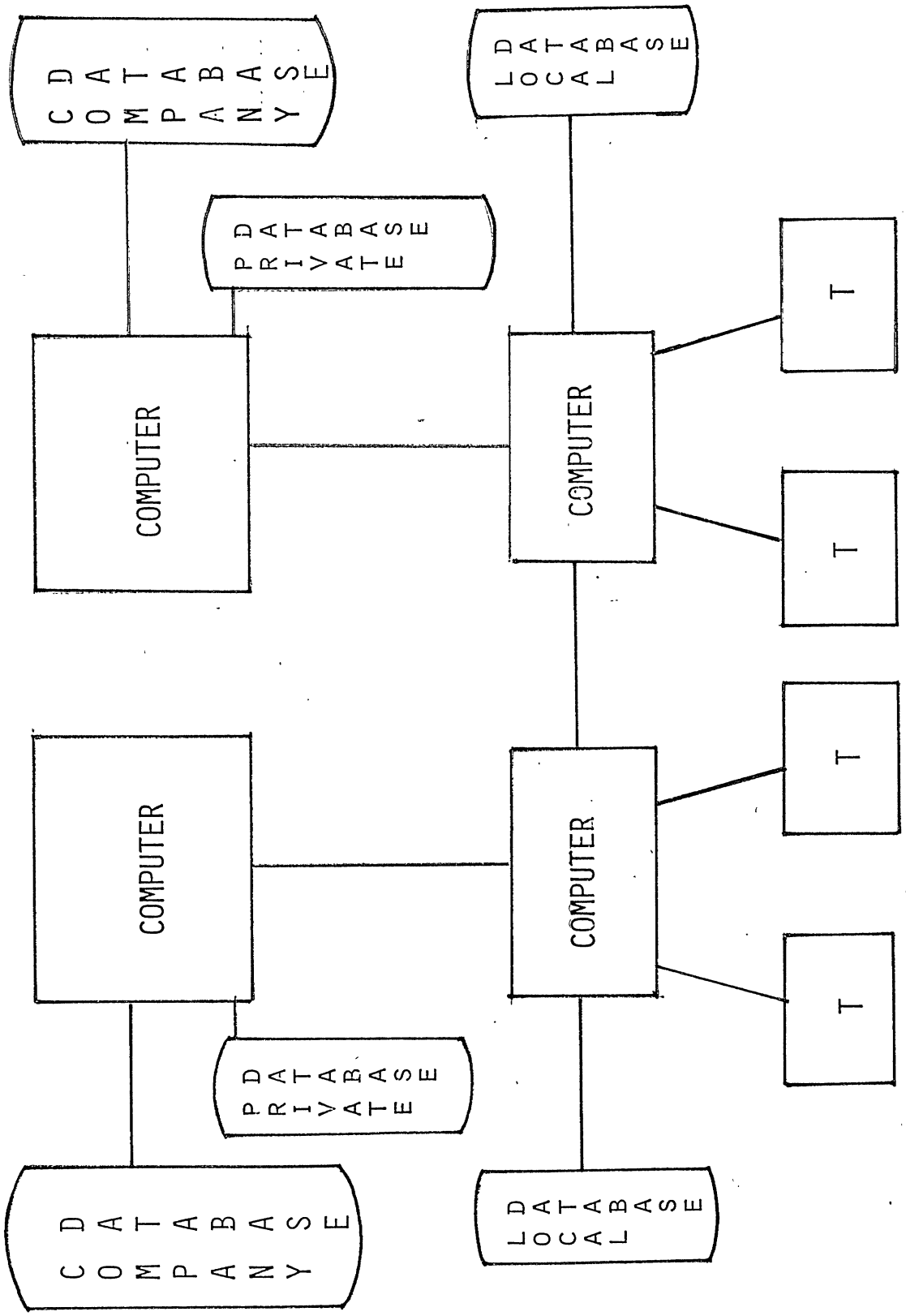


FIGURE 3 PARTIALLY DISTRIBUTED SYSTEM

Applications at one node requiring access to files attached to more than one information processor introduces technological complexities not evident in the previous two definitions. Private files may exist, known only to a specific remote site, or they may exist as skeleton copies of company files residing at other nodes. Existence of both private and company files introduces problems of data consistency, data integrity, and breaches of data security. Sensitive data, if not subject to equally stringent controls at all nodes, may become resident as uncontrolled data in non-local, private files subject to access, which compromises their sensitive nature. The technological resolution of these problems is non-trivial.

As with DDS, partially distributed systems primarily support geographic partitioning. Communications are frequently accomplished through batch processing. Partially distributed systems exhibit an important characteristic of "true" distributed systems in that the processing focus is no longer on a large centralized computer.

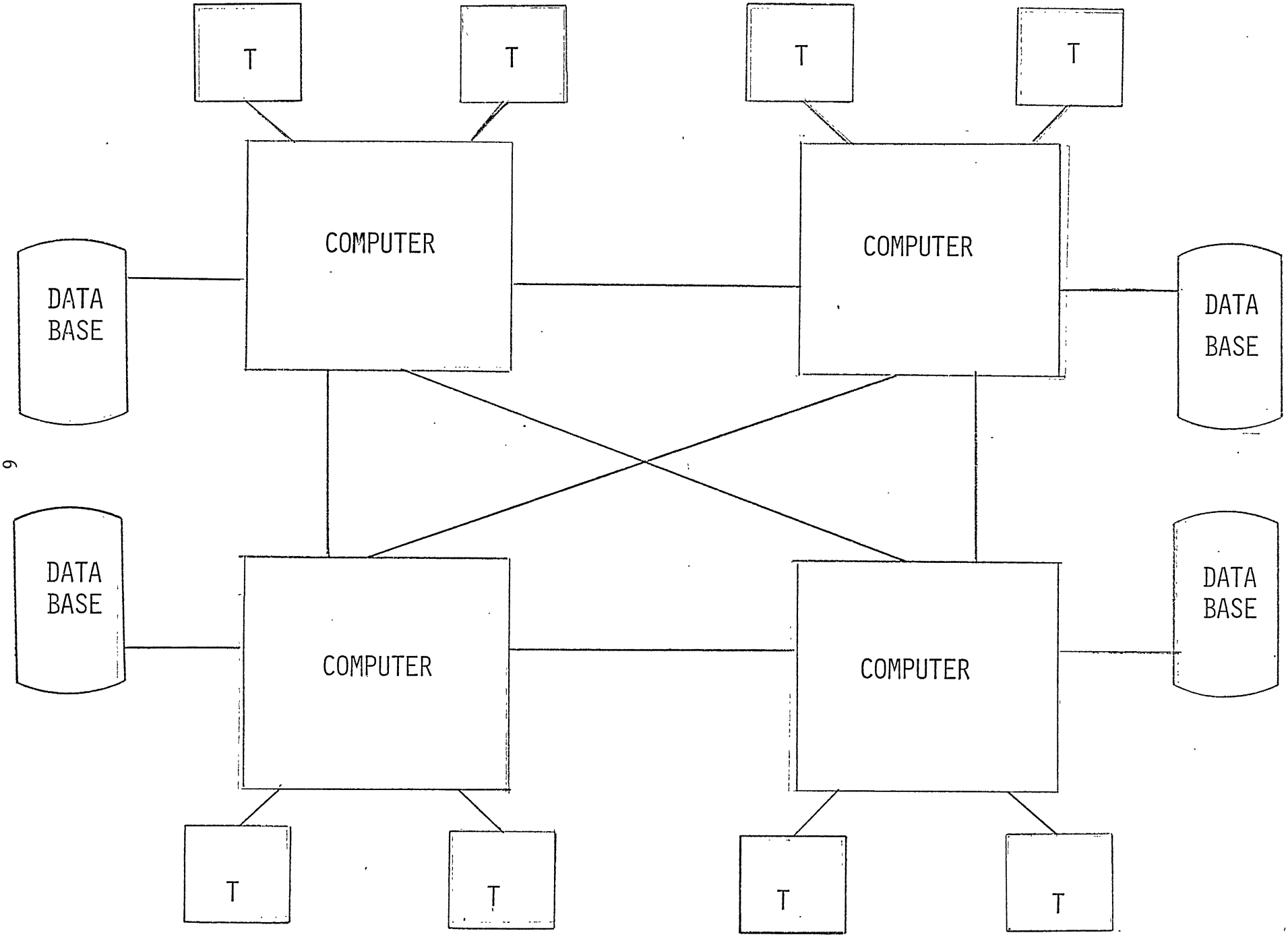
Fully Distributed Systems

A fully distributed system meets the following characteristics

1. Two or more general-purpose processors.
2. A system operating system.
3. Employment of a communications type protocol.
4. Services are requested by name.
5. Non-deterministic resources allocation (Enslow 1976).

Figures 4 and 5 schematically represent the two major configurations identified as fully distributed systems. Many

FIGURE 4 FULLY DISTRIBUTED SYSTEM, WITHOUT HOST



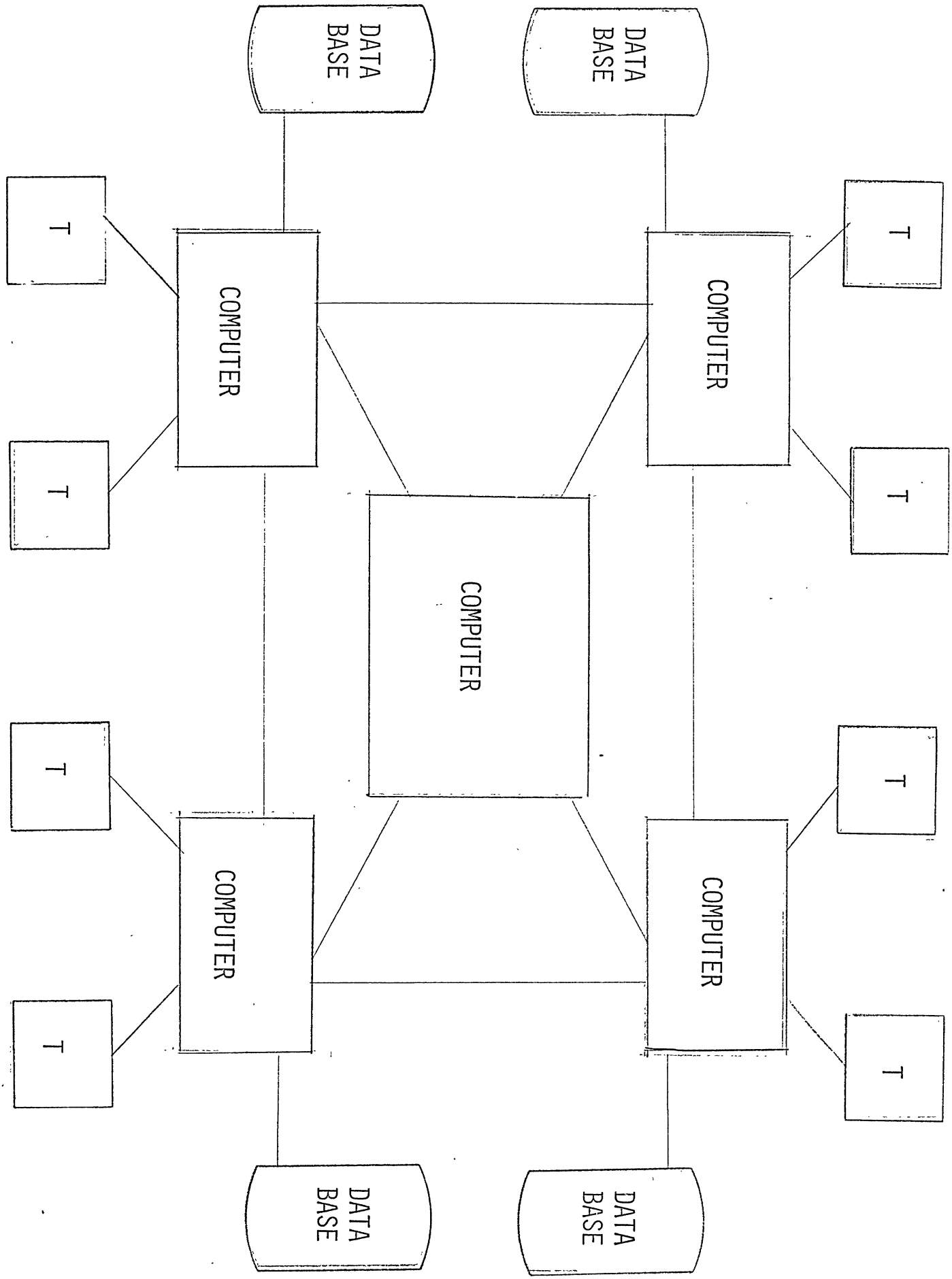


FIGURE 5 FULLY DISTRIBUTED SYSTEM, WITH HOST

authors argue that a host creates a master/slave relationship between the host and the nodes, thereby violating criterion number 3. Other controversies focus on the degree of logical and physical coupling, the degree to which the processing load is shared, and the method of handling concurrent updates. The technological differences between these two configurations have a profound impact on audit trail generation, system generated audit tools, and overall auditability of the system.

Both host and hostless configurations provide on-line interaction among nodes through permanent data communication channels rather than temporary telecommunication links. Physical location of data and application programs are transparent to users. Some system architectures employ dynamic relocation of application programs and/or portions of the data base. Additional complexity results from the existence of multiple copies of the database, either as total duplications, skeleton copies, or a combination of a detailed copy at one node with a variety of summary-type copies resident at one or more additional nodes. The operating system itself may be duplicated at each node or distributed over all or some of the network nodes.

Data are maintained in a relatively live fashion, requiring establishment of protection systems for concurrent updating and lockout protection. Technology to support concurrent access-updates is a major distinguishing characteristic between different fully distributed systems.

Distributed system complexity is compounded by the use of Database Management Systems (DBMSs) (Wilkins 1978). Demands for improved system responsiveness and greater accessibility of data

resources fostered the evolution of distributed systems. Full satisfaction of these demands cannot be effectively accomplished without a union between DBMS and distributed systems. Development of procedures and guidelines for auditing Database Management Systems will have to address both centralized and distributed systems.

Distributed System Summary

The four definitions of distributed systems presented above illustrate a continuum on which such systems lie. To date, users and developers have perceived "distributed" as one of the first two perspectives, because these two configurations represent the majority of distributed systems found in commercial applications.

2. AUDIT PROBLEMS ENCOUNTERED WITH DISTRIBUTED SYSTEMS

Some audit problems will be encountered with the development of any new EDP technology; distributed systems are no exception. The subsequent paragraphs will discuss six sources of audit problems (Table 1) related to distributed systems. Although the discussion context is that of fully distributed systems, some or all of these problems are encountered, at varying degrees of significance, in lower level systems.

The problem sources presented in Table 1 neither originated with nor are they unique to distributed systems. Evolutionary progress of system technology and ever increasing user demands for system access and performance began in higher-level second-generation systems. Distributed systems only serve to magnify their impact.

A. EXPANDED USER BASE	D. ERROR RECOVERY
B. MULTIPLE COPIES OF APPLICATIONS PROGRAMS	E. COMMUNICATION CHANNELS
C. MULTIPLE COPIES OF THE DATABASE	F. WEAK LINK CONCEPT

TABLE 1
SOURCES OF AUDIT PROBLEM

Expanded User Base

Introduction of on-line interactive capabilities inaugurated an irreversible trend toward an expanded user base, both as a data entry source and as a requestor for output. Distributed systems represent significant progress toward placing data resources at the users' fingertips.

Movement of data input responsibility back toward the originator shifts the locus of responsibility for maintaining data input integrity from a centralized keying operation within the EDP department to "system" controls and checks. "System" controls must be sufficient to identify and protect against errors generated by personnel of vastly differing input skills. Proliferation of on-line access heightens user awareness of computer capabilities. Increased awareness in turn leads to demands for even greater service levels, for specialized program support, and for program modifications to satisfy unique user requirements. Therefore program development, control, and maintenance becomes a difficult task and an increasingly greater source for potential errors or inconsistencies.

System exposure and vulnerability are greatly increased as the number of system access points increase. Within a centralized EDP department the potential for system penetration is limited in part by physical security requirements. Dispersed entry points beyond the confines of the EDP department make it much easier for a potential penetrator to have access to the system, experiment, and attempt penetration with a greater degree of anonymity and a lower probability of detection. Penetration also represents a potential threat as the number of users who could purposely or inadvertently

make security breach possible through loss or compromise of user passwords and identification increases. Therefore, not only must system controls be stringent, but also the control parameters must constantly be altered.

Multiple Copies of Application Programs

Multiple copies of application programs are the source of two problem types: program maintenance and program use. Historically, an integral control feature of both maintenance and use has been a centralized EDP function. With limited program copies, centrally located maintenance could be somewhat routine. Existence of multiple copies resident on geographically dispersed nodes dilutes program control spatially if not in fact. Therefore, at a time when program control is most critical and the complexity of program maintenance is increasing, direct control, the vehicle for ensuring program consistency, is decreasing. The potential for either unauthorized program alterations or errors in program maintenance to exist undetected is significantly increased. Surrogate control features must provide the protection formerly relegated to the power of centralized control.

Only authorized users should be allowed program access. Nodal control features must be at least as stringent as those of a centralized system. Where programs are dynamically reallocated, static control features must afford the highest level of protection required to ensure program integrity, or control features must be dynamically expanded as far as necessary to ensure integrity of the reallocated program. The cost of high level static controls or dynamic control allocation is system overhead and response time degradation. Error detection is difficult when programs can be

dynamically reallocated, because the point of error is constantly changing. Once an error is detected the audit trail of that program's activity and the origin of the error may have been lost, making database corrections difficult if not impossible.

Multiple Copies of the Database

Data inconsistency, loss of data integrity, and error perpetration are problems which may result when multiple copies of the database exist.

Supporting live data at each node, possibly in different formats or at different levels of summarization, makes simple, periodic consistency checks impossible. System controls must be established to ensure that each data alteration is properly recorded within each nodal database, and periodic consistency checks must accommodate the complexity created by the variety of formats and levels of detail by which data are stored. Technological problems in supporting concurrent updates of live data within a distributed system have not been solved. Although much of the debate focuses on the accomplishment and sequence of accomplishment for concurrent updates, one of the most critical problems is maintenance of protection and control in the event of system failure during updates. System failure occurring at a point in which some, but not all, nodes have been updated can destroy data consistency. Therefore, the system-generated audit trails must be complete during both the concurrent update process and any subsequent recovery process where such updates have been interrupted by system failures.

Most advanced EDP systems are event-driven. This capability for a single transaction to automatically initiate one or a

multitude of other transactions increases the potential for widespread impact of an erroneous or unauthorized data alteration before the error is discovered. Where the error is a result of system penetration, the time lag incurred in identifying and instituting a search for the error source may permit the perpetrator sufficient opportunity to restore the error source to the proper value. As in the case of program errors, dynamic reallocation of the database or subsets thereof adds significant complexity to error tracing. System-generated audit trails may have been destroyed before error detection and initiation of tracing occurs.

Multiple database copies expand the points at which data errors (primarily resulting from input) or penetration may occur. As discussed in conjunction with multiple copies of application programs, high-level static control features must exist at nodes or be dynamically expanded as necessary to accommodate the requirements of reallocated data. Even temporary residence of sensitive data at a node without sufficient control features potentially compromises such data.

Error Recovery

Error recovery is certainly not a new problem source for EDP systems. With distributed systems, failure of a node not only impacts primary users and systems of that node but, potentially, the entire network. Complete node isolation results in temporary loss of access to data and programs resident at that node. System controls must ensure that transactions requiring or interfacing with the isolated node are not lost. Networks in which another node "stands-in" for the failed node must accommodate complete and accurate transfer of application programs and the database. The

"stand-in" node must exhibit system controls necessary to ensure the proper level of protection required of the transferred information. Furthermore, the "stand-in" node must be able to absorb the increased workload without itself overloading, failing, and creating a domino failure among the network nodes.

Geographic nodal separation complicates recovery procedures. Recovery must be accomplished by a specialized team from the EDP department, or each node location must retain qualified EDP personnel on-site. A specialized team introduces recovery delays. Decentralized recovery requires adequate training, system checks, and periodic observations to ensure compliance with EDP department recovery standards by each node location. As with any recovery procedure, data integrity of the restored node must be assured. All updates or transmissions pending at or subsequent to the time of failure must be accomplished.

Communication Channels

Communication channels are the source of two major problems: channel failure and penetration. Communication channels potentially represent the most vulnerable element of a distributed system. Interruption may occur as a result of "physical exposure" either through natural causes or externally inflicted damage. Data loss or contamination may result from interference during transmission or receipt. System controls must ensure transmitted data integrity.

Penetration of communication links may transpire without direct detection. Links may be tapped passively or data transmissions may be actively altered by overlapping a signal on transmissions and stripping-off that same signal on receipt acknowledgement transmissions. Protection against passive tapping must be

accomplished through physical security of the communication link and encoding of transmissions. Protection from active penetration requires system controls external to the communication channel.

Weak Link Concept

The weak link concept is a logical extension of the preceding discussions of problem sources; it is, however, important to give recognition specifically to the fact that the network is only as strong as its weakest link. Any location which fails to exercise the requisite standards of physical or internal control, fails to adequately train its personnel, or fails to exercise a reasonable measure of constraints over its user demands jeopardizes the entire network.

3. RECOMMENDED AUDIT TECHNIQUES

As was noted in the preceding discussion, most of the problem sources did not originate with nor are they unique to distributed systems. Therefore, it should not be unexpected that the recommended techniques for auditing distributed systems are not entirely revolutionary ideas. The value of the proposed recommendations derives not from their uniqueness, but from their representation of an iteration of specific actions which should be included in audit procedures.

The importance of the alliances between EDP personnel and independent and internal auditors is critical to the auditability of distributed systems (Carney and Merten 1978). The listing of proposed techniques contains steps which should be conducted in addition to current audit procedures. It is not, however, intended to be exhaustive, nor is it expected or necessary that their

applicability be restricted to the categories in which they are presented.

Expanded User Base

An expanded user base should impose no new audit requirements. Interactive users existed in many environments before the advent of distributed systems. However, definitive guidelines as to procedural audit requirements have not been issued; therefore it is recommended that auditors incorporate the following into existing procedures:

1. Compare data transmitted with that received based on system generated activity logs.

2. Specifically examine transactions initiated outside of normal working hours, including those initiated during designated meal hours.

3. Specifically examine transactions initiated at authorized terminal locations other than the one most frequently used by a given user.

4. Examine files for which there have been unauthorized access attempts.

Multiple Copies of Application Programs

Resolution of this problem can be achieved through the use of some of the techniques which were noted in a recent survey as being seldom employed (Perry and Warner 1978). The most critical element proposed is the assurance that each node exhibit the requisite control features compatible with the programs resident at that node.

Recommended techniques for this section are as follows:

1. Review control systems in place at each node in conjunction with programs currently residing at that node.

2. Examine the audit trail of program reallocations.

3. On a routine and surprise basis perform a bit-by-bit comparison of object programs actually in use with the library copy. Each sample should include selected sensitive programs.

4. On a routine audit basis, randomly select programs for flowchart generation and review against existing documentation.

5. Compare changes found in items 3 and 4 with recorded authorizations.

Multiple Copies of the Database

Multiple copies of live data require that auditors audit live transactions which requires a capability not incorporated into most generalized software packages. Effective audit of live transactions will require increased utilization of surprise audit techniques; periodic scheduled reviews are inadequate to accomplish this task. Again the infrequently employed techniques must be elevated in importance (Perry and Warner 1978). The following procedures should be implemented:

1. On both a routine and surprise basis
    a. Use tagged transactions.
    b. Compare audit logs to verify that the data sent is the same as the data that is received.
    c. Compare randomly selected files resident on multiple nodes.
2. Compare location directories with the database actually resident at each node.

3. Subsequent to system recovery actions, compare selected files (see error recovery recommendation).

Error Recovery

Observation of actual error recovery should be part of every audit. Ineffective implementation of recovery procedures despite excellently written standards can result in vast amounts of undetected data loss. The ability of the auditor to initiate systems crashes would require client approval. Such crashes would have to be extremely well planned and coordinated with selected client personnel to preserve both the surprise element and the integrity of the database. Observation of recovery techniques imposes problems for the auditor, primarily because of the multiplicity of potential recovery points and the geographic dispersion of the nodes. Despite the executional and organizational difficulties noted, the following recommendations are considered important audit procedures:

1) Auditors should initiate system crashes on a surprise basis.

2) Audit and review of each node's recovery actions.

Communication Channels

Evaluation of communication channels may require a high degree of technical expertise. A viable option would be for auditors to employ communications technicians or to engage technical consultants as a matter of routine. The audit procedures should be included at a minimum the comparison of data transmitted with that received (on a routine and a surprise basis).

Weak Link Concept

Incorporation of the above recommendations into current audit procedures could make a significant contribution to ensuring that the weakest link in the network will at least satisfy the minimal acceptable standards. Such assurances are necessary if an organization is to place any reliance upon the financial data generated by a distributed system.

4. CONCLUSIONS

Four definitions of distributed systems and their impact on the audit function have been discussed. Although distributed processing represents the fourth generation of information systems, evolution of a fifth generation may already be in a conceptual development stage. The computer must be perceived as a tool for auditing rather than just an element of client's financial system to be audited. Increased existence of live data and decreasing audit trail half-life may require audit re-orientation. Distributed systems represent both an opportunity and a threat with respect to interval control of an organization. A cooperative effort among users, data processing, and auditing is needed to ensure a favorable outcome.

BIBLIOGRAPHY

Booth, Grayce M. "The Use of Distributed Data Bases in Information Networks." Advances in Computer Communications, ARTECH House, 1974.

Brown, Richard. A History of Accounting and Accountants. A.M. Kelley, New York, 1968.

Carney, R. "Distributed Data Systems." EDP Analyzer, June 1976.

Carney, D., and Merten, A. "Need for an Auditing Renaissance: The Generation Gap Between Auditing and Information Systems." Unpublished paper, University of Michigan, 1978.

Commerce Clearing House. Topical Law Reports: Accountancy Law Reporter. (Chicago: Commerce Clearing House) 2d ed., vol. 1 & 2.

Enslow, P. "What Does Distributed Processing Mean?" Computer Architecture News, vol. 5, no. 5, December 1976.

Perry, W.E., and Warner, H.C. "Systems Auditability: Friend or Foe" Journal of Accountancy, February 1978.

Wilkins, D.G. "Data Base: A New Frontier in Auditing" Unpublished paper, University of Michigan, 1978.