韓國 軍事運營分析 學會誌
第12卷 第2號, 1986.12.

Database Security and Integrity for the Republic of Korea Military

Kim, Se Kyung
Lim, Seong Nam

I. Introduction

Information processing has become a critical support system in the daily operations of government, business, and industry. With the importance of information, computer systems, data bases, and computer-communication networks are critical nerve systems of modern organizations.

Computer security was first recognized as a problem in the mid-1960s when timeshared and multiprogrammed computer systems emerged. It was shown that the then existing operating systems were unable to prevent determined users from gaining unauthorized accesses to or modifications of other users' data or programs, or from intercepting data on communications lines between terminals and computers, or between computer systems [Ref. 1, Ref. 2]. The problem was especially acute for computer applications in national defense systems [Ref. 3]. Since then, research and development in the computer security area has grown steadily, but solutions have been difficult to produce [Ref. 4].

With the problem of security, the continued successful operation of an enterprise demands that:

(1) confidential data is available only to authorized persons, so that privacy requirements are satisfied and enterprise secrets guarded;

(2) the data accurately reflects the state of the enterprise; that is, the data is protected against either malicious or accidental destruction or corruption.

* ROKN Headquarters

Just as important as the economic value of information is the privacy of individuals. The effect of disclosing or altering information about individuals ranges from annoyance to endangering their lives. Privacy laws may require, for example, that personal information be used only for the original purpose for which it was collected, or that the accuracy of the information be maintained. Other laws are intended to protect customers and stockholders against loss due to lack of control of information resources. Thus there are also legal reasons for an enterprise to maintain the security and integrity of its information.

Typically a database contains data of various degrees of importance and levels of sensitivity. This data is shared among a wide variety of users with different responsibilities and privileges. It therefore becomes important to restrict users of the database to those portions of the total data that are necessary for their activities. Additionally, more control is needed over what changes a user can make to data because of the many ways these changes can effect other users of the database.

Another characteristic of databases is that a change to a data value destroys the old value. This means that new recovery techniques are needed to restore the database after system and program failures or errors.

Finally, many databases are online, accessed by multiple users concurrently. Special mechanisms are needed in this situation to maintain the integrity of the database and the consistency of information that is retrieved.

Although security and integrity measures within a database management system are necessary, these alone are not sufficient. The underlying systems, such as the operating system and the computer hardware, must also have security procedures external to the computer system, such as locks on the computer-room door and fireproof safes for storing copies of the database.

Historically man's quest for obtaining information and data has been the basis for innumerable tales of intrigue, deception, and ingenuity. According to Kann and Prywes [Ref. 5 and 6], the first data handling revolution began around 1650 with the institution of regular intercity postal services. Shortly thereafter, government groups often called "black chamber operation" were organized to illegally collect the information. They would intercept the mail, extract useful information, re-seal the letters, and send them on without the sender's or receiver's knowledge. The second data handling revolution began with the introduction of the telegraph about 1850. Again government organized groups, as well as commercially sponsored teams, were used for the illegal interception, decoding, and distribution of telegraphic messages. The third data handling revolution commenced around 1895 with the introduction of the radio. Presently we find ourselves a few years into a major technological revolution in data handling involving computers. Historically then, it is not surprising to find individuals and organizations involved in the work of illegally obtaining, manipulating, destroying, or in some way compromising computer based data.

A close examination of data security requiretents indicates that they are dependent on the specific type of threat posed to the system. There are three general classes of threats: unintentional, deliberate passive, and deliberate active.

Unintentional threats are those that arise from hardware and software failures and user errors which allow unauthorized but inadvertent access to files or programs.

Deliberate passive threats are caused by electromagnetic radiation from the computer hardware and communications equipment. Passive methods include wiretapping and monitoring of electromagnetic emanation.

Deliberate active threats are attempts to enter the system to obtain data from files or to interfere with data files or the system. Examples of this type of threat are (1) using legitimate access to ask or obtain unauthorized access (browsing), (2) masquerading as a legitimate user, (3) using access to the system as support personnel (systems programmers, operators, hardware maintenance, management) to obtain data or create trap doors into the system, (4) tapping into remote terminals to receive "piggy back" entry with an authorized user, (5) between lines entry, and cancellation of user's sign off signals to continue operation.

These threats are nearly the same for all systems, differing primarily in the degree to which system design features allow exploitation. This potential for exploitation is created at each point where a user interacts with the system. Since the security requirements depend on the threat of exploitation and the threat of exploitation in turn depends on the particular system access point, the key to specifying the security requirements for a system lies in an examination of the system's accessibility.

II. Data Security and Integrity Implementation

For the various kinds of data security and integrity methodologies, please refer a list of references 7 through 28. In this chapter we will investigate implementation issues in terms of ORACLE and now security and integrity works for a prototype Korean military personnel database. The ORACLE relational database has been considered as a promising candidate for use with the Korean military personnel database.

A Data Security

To secure a user's database, ORACLE will keep a user's data private unless the owner allows access privileges to other users. On the other hand, by allowing some access privileges, other users can share the owner's database. The following methodologies show now ORACLE protects the personnel database.

1. Identification

To be identified by ORACLE before performing any operations in ORACLE, we must provide a USERID and PASSWORD after logging on.

[Example 1] : Use the UFI command CONNECT to log on to ORACLE as user LIM with a password of AFLIM.

UFI> CONNECT LIM/AFLIM
Connected.

On the other hand, ORACLE rejects unauthorized personnel who try to access the database without an acceptable password.

[Example 2] : Try to log on with a user name of LEE and a password of NAVY.

UFI > CONNECT LEE/NAVY
ERROR: Invalid username/password–logon denied.
Warning: You are no longer connected to ORACLE!

2. Sharing Data with Other Users

An authorized user who creates a table in ORACLE becomes the owner of that table. The owner gives or takes away access privileges to or from the other persons by using the following SQL commands:

GRANT – gives other users access to his/her tables
REVOKE – takes away other user's access to his/her tables

3. Granting Rights to Other Users

The Grant command is made up of three basic clauses:

GRANT – a function
ON – a table or view
TO – a user

[Example 3] : Have user LIM (the currently logged on user) grant the SELECT privilege on the PERSON table to a user named KIM.

UFI > CONNECT LIM/AFLIM
Connected.

Use the command SELECT to display the PERSON table which was created by the user named LIM.

UFI > SELECT
2 FROM PERSON;

SN	RANK	NAME	MOS	CD	CT	MS
234567	COLONEL	CHA, S.H.	111	11-MAR-65	1	1
245678	MAJOR	KIM, K.S.	121	20-APR-66	1	1
245001	MAJOR	PARK, S.U.	130	21-SEP-66	1	2

Next, give the SELECT privileges on the PERSON table to a user named KIM.

```
UFI > GRANT     SELECT
  2   ON        PERSON
  3   TO        KIM;
```

Grant succeeded.

To check whether the SELECT privilege was granted to KIM, we will command as follows:

[Example 4] : Have KIM SELECT data from the PERSON table.

```
UFI > CONNECT     KIM/SEKYUNG
```

Connected.

```
UFI > SELECT
  2   FROM        LIM. PERSON;
```

SN	RANK	NAME	MOS	CD	CT	MS
234567	COLONEL	CHA, S.H.	111	11-MAR-65	1	1
245678	MAJOR	KIM, K.S.	121	20-APR-66	1	1
245001	MAJOR	PARK, S.U.	130	21-SEP-66	1	2

To display the table PERSON which was created by LIM, KIM, must prefix "LIM" as "LIM PERSON". By this procedure the PERSON table which was created by LIM is displayed as above

4. Granting Multiple Privilege to Multiple User

By using the GRANT command, the owner car, give some privileges to more than one user.

[Example 5] : Give both KIM and DOIK the INSERT and UPDATE privileges on the PERSON table

```
UFI > GRANT     INSERT, UPDATE
  2   ON        PERSON
  3   TO        KIM, DOLK;
```

Grant succeeded.

5. Creating Synonym for Table

To change the short and simple table name from the previous long name, we use the following command.

[Example 6] : Create a synonym named P to be used in place of LIM. PERSON.

```
UFI > CREATE     SYNONYM P
  2  FOR         LIM. PERSON;
```

Synonym created.

[Example 7] : Query the PERSON table using the synonym P.

```
UFI > SELECT
  FROM          P;
```

SN	RANK	NAME	MOS	CD	CT	MS
234567	COLONEL	CHA, S.H.	111	11-MAR-65	1	1
245678	MAJOR	KIM, K.S.	121	20-APR-66	1	1
245001	MAJOR	PARK, S.U.	130	21-SEP-66	1	2

6. Attempted Security Violation

Since the owner of the table PERSON whose name is LIM didn't grant the DELETE privilege, KIM cannot delete the PERSON table. The following example shows the violation and its result.

[Example 8] : Have KIM attempt to DELETE a row from the PERSON table.

```
UFI > DELETE     FROM LIM. PERSON WHERE SN = 245001;
DELETE     FROM LIM. PERSON WHERE SN = 245001
                    *
ERROR at line 1: table or view does not exist
```

Because KIM is not authorized to delete rows from the person table, he is told that the table does not exist.

7. Granting All Privileges to a User

A user can grant all privileges on a table by specifying the key word ALL in place of the list of privileges.

[Example 9] : Grant all privileges on the PERSON table to KIM.

```
UFI > CONNECT    LIM/AFLIM
Connected.
UFI > GRANT      ALL
  2   ON         PERSON
  3   TO         KIM;
Grant succeeded.
UFI > CONNECT    KIM/SEKYUNG
Connected.
UFI > DELETE     FROM LIM. PERSON WHERE SN = 245001;
                 1 record deleted.
UFI > SELECT
  2   FROM       LIM. PERSON;
```

SN	RANK	NAME	MOS	CD	CT	MS
234567	COLONEL	CHA, S.H.	111	11-MAR-65	1	1
245678	MAJOR	KIM, K.S.	121	20-APR-66	1	1

If someone (other than creator) tries to erase PERSON table, ORACLE will deny as follows:

```
UFI > DROP     TABLE PERSON;
DROP     TABLE     PERSON
Error at line 1: table or view does not exist.
```

Therefore only the creator of the PERSON table (LIM) can drop the table.

8. Revoke a Privilege

To revoke the privileges which were granted by the creator from the grantee, the REVOKE command is used as follows:

[Example 10] : Revoke from KIM the right to INSERT into the PERSON table.

```
UFI > REVOKE     INSERT
  2   ON         PERSON
  3   FROM       KIM;
Revoke succeeded.
```

9. The Data Base Administrator

When the ORACLE database is first created, there is only one user authorized to log on. In the ORACLE database environment at NPS, Mrs. Jeannie Bowers is authorized as database administrator. We can be granted full DBA authority by using her user name (e.g., JEANNIE) and password (e.g., BOWERS) at which time we can create other users.

[Example 11] : Create a new user named LIM with DBA authority.

UFI > CONNECT JEANNIE/BOWERS

Connected.

UFI > GRANT DBA TO LIM IDENTIFIED BY AFLIM;

Grant succeeded.

This command must be implemented as above. We couldn't do so since JEANNIE BOWERS naturally didn't want to disclose her username and password.

10. Changing User Password

The user who has DBA authority has to change his password to prevent other users misusing his DBA authority after disclosing his username and password.

[Example 12] : Change the password of the user JEANNIE from BOWERS to DOLPHIN.

UFI > GRANT CONNECT TO JEANNIE IDENTIFIED BY DOLPHIN;

Grant succeeded.

[Example 13] : Try to log on as user JEANNIE with a password of BOWERS.

UFI > CONNECT JEANNIE/BOWERS
ERROR: Invalid username/password–logon denied.

Warning: You are no longer connected to ORACLE!

[Example 14] : Try to log on as user JEANNIE with a password of DOLPHIN.

UFI > CONNECT JEANNIE/DOLPHIN

Connected.

11. Revoke a User's Privilege to Use the Database

Any user with DBA authority can revoke another users' authority to access the database.

[Example 15] : Log on as DBA LIM and revoke user KIM's CONNECT authority.

UFI > CONNECT LIM/AFLIM

Connected.

UFI > REVOKE CONNECT FROM KIM;

Revoke succeeded.

[Example 16] : Try to log on as KIM.

UFI > CONNECT KIM/SEKYUNG
ERROR: Invalid username/password–log on denied.

Warning: You are no longer connected to ORACLE!

Through the various kinds of ORACLE security methodelogies which were mentioned above, we found facilities to prevent incorrect data (i.e., by ORACLE's IAF function) from being in a database and to prevent the reading of data that should not be disclosed to unauthorized personnel in Korea's military environment.

12. Evaluation

In this section we will investigate what ORACLE supports for data security and what relationship ORACLE has to the protective mechanisms which were discussed in chapter III.

a. Modes of Privilege

We can protect the database from the application program by using rings of privilege. Application program X has direct access to its private files A and B but not to the database. However, it can call the SELECT procedure of the DBMS, which does have READ access to the database. Also, application program X can access the database only indirectly through calling the DBMS, while the DBMS cannot access File A and B belonging to application X. In the ORACLE system, the GRANT command grants identical privileges to authorized users. Therefore ORACLE supports the rings of privilege concept within its DBMS environment.

b. Memory Protection

One simple form of memory protection uses "locks" and "keys". Memory is divided into blocks, and a lock in assigned to each block. At any moment, the CPU has a "protection key" that is part of the processor status, and writing is allowed only if the protection key matches the lock. The number of different keys is often quite limited and this restricts the flexibility of the lock-and-key mechanism. ORACLE provides a LOCK command to prevent different processes (users) from trying to update the same record with different values. ORACLE prevents conflicting access to the database by acquiring locks on the data accessed by local units of work.

c. File Protection

Passwords are normally supplied by the program, which means that, since the password probably is contained in the program, the program must then be protected. One serious problem with passwords is that there is no efficient way to withdraw rights from one of many users. But in ORACLE, use of the REVOKE command by users who have DBA authority eliminates this kind of problem. Another serious problem is keeping the password secret. The password for a DBMS file will usually either be coded into the DBMS program or entered by an operator or initial invoker of the DBMS system. Even in ORACLE, this problem cannot be handled and the possibility of password leakage exists. For perfect secrecy, we could apply fingerprint as well as password. This is currently not available in ORACLE.

d. Conclusion

So far, we have studied protective mechanisms for data security in theoretical terms. Aditionally, we have investigated what ORACLE supports for database security in terms of real world problems to try to correlate theory and practice. As compared and mentioned in the "evaluation" part, ORACLE supports some security techniques well while not supporting others such as encryption/decryption. Overall, the security mechanisms in ORACLE are recommended as adequate methodology for the personnel database application in Korea's military environment.

B. Integrity

In this section, we are going to analyze the ORACLE DBMS with respect to data integrity using the PERSON table already referenced and the PER10 view which will be created. We are also going to evaluate which integrity method is best for the personnel management DB design.

1. Concurrency

Several users of ORACLE may share the same executable code and the same data via concurrency mechanisms. Since two or more users may want to read or update the same record in the same file, a locking mechanism is required to prevent the same record from being updated with different values. ORACLE has a capability which prevents conflicting access to the database by requiring locks on the data accessed by logical units of work. Although user programs don't request locks, ORACLE locks the necessary data automatically. A user will implicitly lock a row for shared access during any SELECT statement. Since a person might be updating another row located in a different position of the same table, the data in the table may not be consistent. ORACLE can lock out updaters to that table whenever a user wants consistent data.

LOCK TABLE tablename IN SHARE MODE

UFI > LOCK TABLE PERSON IN SHARE MODE;

Table(s) locked.

This command does not allow other users to update the table until the current user is finished but does allow concurrent reading of the table.

For the entire logical unit of work, the UPDATE or INSERT command will lock the table implicitly. So after the exclusive reader commits his data, consistent readers can be allowed access to the table. Since this locking mechanism is necessary for COMMIT/ROLLBACK recovery (i.e., the COMMIT/ROLLBACK commands are implemented in ORACLE by journaling previous images of changed pages of the database.), We must be very careful to force writers to commit their work quickly. The following command shows locking a table for exclusive use.

LOCK TABLE tablename IN EXCLUSIVE MODE

```
UFI > LOCK TABLE PERSON IN EXCLUSIVE MODE;
Table(s) locked.
```

We must use the FOR UPDATE clause in the SELECT statement to lock the minimum amount of rows at one time. The rows which are selected by the SELECT statement will be locked by the FOR UPDATE command.

```
UFI > SELECT     RANK, NAME, CD
  2   FROM       PERSON
  3   WHERE      SN = 245678
  4   FOR UPDATE OF RANK;
```

RANK	NAME	CD
MAJOR	KIM, S.K.	20-APR-66

```
UFI > UPDATE   PERSON
  2   SET      RANK ='L. COL.'
  3   WHERE    SN = 245678;
1 record updated.

UFI > SELECT   FROM PERSON;
```

SN	RANK	NAME	MOS	CD	CT	MS
234567	COLONEL	CHA, S.H.	111	11-MAR-65	1	1
245678	L. COL.	KIM, S.K.	121	20-APR-66	1	1
245001	MAJOR	PARK, S.U.	130	21-SEP-66	1	2

2. Deadlock Detection

If the data have been locked by two users each of whom is waiting for the other to finish accessing the data, a deadlock takes place. In this situation, ORACLE detects and rolls back the logical unit of work with the smallest number of blocks previously imaged. ORACLE has the ability to restore all the changes made to the database and to release all blocks owned by the logical unit of work that has used the fewest resources. This allows the other logical unit of work to continue. If the logical unit of work is rejected, it will receive an error message (i.e., deadlock has been detected and all changes have been rolled back). On the other hand, the logical unit of work that has been rolled back can retry the commands or exit, and try again at a later period of time. Since many dictionary tables must be updated via DDL commands, deadlock can be detected easily in DDL mode. When a deadlock is detected, a DDL operation is rolled back and the following message issued:

DDL operation and resource not available

3. Auditing

ORACLE provides a variety of functions with the auditing mecnanism. These include: (1) ORACLE debugging. (2) Audit of attempted security violations. (3) Accounting/Statistical information. (4) Audit trail. When each call to the program that interfaces to a common journal file is posted, the auditing facility is implemented. To reproduce it, enough information is stored at a later time. We need to preserve in a relatively safe place, e.g., on a tape or disk, a history, called a journal or log, of all changes to the database since the last backup copy was made.

The journal file is created at ORACLE initialization time after which it may be processed by the program ODD (ORACLE Dynamic Debugger). The journal file is subsequently closed at ORACLE shutdown time. To use the audit file, we must shut down ORACLE using IOR SHUT and rename the audit file before analyzing the data with the ODD command.

Format is: ODD [auditfile] [option] [-p]

Options are:

D [seq#]	Display process starting at [seq#]
DA	Display All function calls
DL	Display Logins only
S	Summarize function calls
SV	Display security Violations
X [seq#]	Xecute process starting at [seq#]
XA	Xecute All runction calls

-p will display comments, SQL statements variables.

4. Backup and Recovery

a. Backup

By using the EXPORT (EXP) function, we can save a single user's data in a file. EXP will unload one or more tables and/or views of a database (i.e., ORACLE allows us alternative views of the same data stored in the database. Views are "windows" through which we see data that is stored in tables.), will display the names of unloaded tables and views, and will display the row counts for each unloaded table. Example of a View is:

```
UFI > CREATE   VIEW    PER10 AS
  2            SELECT  SN, RANK, MOS
  3            FROM    PERSON
  4            WHERE   CT = 1;
View created.

UFI > SELECT
  2   FROM     PER10;
```

SN	RANK	MOS
234567	COLONEL	111
245678	L. COL.	121
245001	MAJOR	130

The EXP format is as follows:

EXP [dump file] [user/password] – switches

Next, EXP tables (switches) are:

A Audit the export.

Cn Context area size (followed by a number from 1-128, default is 16).

D Export dictionary information only.

E Export in EBCDIC character set format (ASCII is the default).

T Select tables to be exported.

V Select views to be exported.

? Display the EXP internal version number.

The default EXP backs up all tables and views. EXP can only export data one user at a time.

EXP example:

$ EXP PERSON. exp kim/sekyung –tv
Export ORACLE V3.1.0 24-May-1983
File –DUAO: [KIM] PERSON. EXP Exported by KIM on May 8, 1984 08:40:10 pm.

Table : person
Table : salary
Table :
View : per 10
View :
Tables :
PERSON 3 rows
SALARY 3 rows
Views :
PER10
$

b. Recovery

The data of a single user may be restored from an exported file using IMPORT (IMP). A table cannot be created during IMP because processing continues with the next table. If a record cannot be accepted, that record will be displaved and processing will continue with the next record. If a view definition is rejected, it will be displayed. Example is:

$ IMP person. exp kim/sekyung –tv
Import ORACLE V3.1.0 24-May-1983

Can't Import from EXP version: Egwaqp'VCc'xr&e&sKpq
$

IMP will display the names of the tables and views being reloaded. For each table reloaded it will also display the number of rows inserted. If the number of rows actually loaded is different than the rows read, this is indicated with a count of the rows read.

We will deal with IMP format, tables, and an example. First, the IMP format is as follows:

IMP [dump file] [user/password] – switches

Next, IMP tables (switches) are:

A Audit the import.
Cn Context area size (followed by a number from 1-128, default is 16).
D Display contents of the export file without importing.
I Import even if the table creation fails.
T Do not import tables.

V Do not import views.

? Display the IMP internal version number.

IMP example:

$ IMP person. exp kim/sekyung –tv
Import ORACLE V3.1.0 24-May-1983

original export file name: –DUAO: [KIM] PERSON. EXP
export performed by user: KIM
exported under : ORACLE V3.1.0 24-May-1983
database was exported on: May 8, 1984 08:40:10 pm.
Tables:
PERSON : 3 rows read
SALARY : 3 rows read
Views:
PER10
$

5. Evaluation

In this section we will evaluate the relationship between ORACLE support for data integrity and the data integrity issues which were discussed in Chapter IV.

a. Concurrency

ORACLE's concurrency allows several users to share the same executable code and data in a consistent manner. The use of locks minimizes the possibility of inconsistent data undermining the accuracy of the data.

b. Backup and Recovery

For backup and recovery, ORACLE provides the EXP (unloading) and IMP (reloading) utilities that allow a user to unload information from ORACLE onto a sequential file and later reload that information into a database. We think ORACLE provides essential backup and recovery to safeguard against crashes in the personnel database application.

c. Auditing

Auditing wasn't fully discussed in this paper, but it is nevertheless a useful security/integrity feature provided by ORACLE.

d. Conclusion

So far, we have investigated what ORACLE supports for database integrity. Through the evalua-

tion, we have found that locking, auditing, and backup and recovery mechanisms within ORACLE are effective integrity methods for the ROK military personnel DBMS.

In addition to the integrity mechanisms discussed so far, there is the key question of whether ORACLE has the ability to check whether data inserted into a table falls within a prespecified range of values. As an example, assume we have the data item MOS. Can we specify range of between 0 and 500 so if someone tries to input a value of 600 for MOS, ORACLE will print an error? ORACLE's IAF (Interative Application Facility) function has a "Screen Processor" (i.e., Design Screen, Compile Screen, and Process Screen) which allows most types of validation to ensure data integrity. Specifically, it uses "Lowest Value" and "Hignest Value" options to implement a range of values. If the user inserts a value out of range, ORACLE will issue an error message: illegal value, out of range.

Overall, ORACLE's integrity methods are sufficient and recommended as the most appropriate methodology for the ROK military personnel DBMS.

III. Conclusions

This thesis has focused on the ROK military personnel DBMS, however its findings are applicable to all departments of the ROK military needing DBMS security and integrity. In particular the implementation of security and integrity methods using ORACLE results in a more effective and timely presentation of all required personnel information. We believe that these methods can increase the capability of personnel management's security and integrity.

What is the future role of database security and integrity in the ROK military? Database security and integrity will continue to be recognized as an important goal because: (1) new users will demand new database technologies. (2) new hardware and software technology will eliminate old problems while introducing new ones. (3) new solutions will be discovered with continued research on database security and integrity.

Additionally, hardware developments will evolve toward maximizing protection from data misuse or theft by important advances of technique. Greater reliance can be expected on high level languages, and these languages will also provide better support of security and integrity.

We believe that database security and integrity can serve as a prototype from which the ROK military can secure existing systems and add further applications such as encryption and decryption methods.

References

1. Petersen, H.E., and R. Turn, "System Implications of Information Privacy, Proceedings of the 1967 Spring Joint Computer Conference, AFIPS, Vol. 33, pp. 291-300.
2. Ware, W.H., "Security and Privacy: Similarities and Differences, Proceedings of the 1967 Spring Joint Computer Conference, AFIPS, Vol. 30, pp. 287-290.
3. Ware, W.H., (Ed.), Security Controls for Computer Systems, R-609, The Rand Corporation, Santa Monica, CA, February 1970 (Reissued in October 1979 as R-609-1).
4. Tangney, J.D., History of Protection in Computer Systems, MTR-3999, The Mitre Corporation, Beaford, MA, 15 July 1980.
5. Kahn, David, The Code Breakers, Macmillan, p. 163-213, 1967.
6. Prywes, Noan S., "Some Problems and Consideration of Computer Security, Naval Ship Research and Development Center (NSRDC), Proceedings of The Conference on Secure Data Sharing, p. 144-152, August 1973.
7. CODASYL Data Description Language. NBS Handbook 113, Nat. Bureau of Standards, Washington, D.C., June 1973.
8. Granam, G.S. and Denning, P.J., "Protection Principles and Practice", Proceedings of the 1972 Spring Joint Computer Conference, AFIPS, Vol. 40, p. 417-429, 1972.
9. Graham R.M., "Protection in an Information Processing Utility", Com. ACM 11, 5 (May 1968), p. 365-369.
0. Schroeder M.D. and Saltzer J.H., "A Hardware Architecture for Implementing Protection Rings". Comm. ACM 15, 3 (March 1972), p. 157-170.
11. IBM Corporation, "OS/VS2 Access Method Services". Form No. GC26-3941, San Jose, Calif. 1978.
12. IBM Corperation, "OS/VS2 MVS Resource Access-Control Facility (RCAF), General Information Manual". Form No. GC28-0722-4, Pougnkeepsie, N.Y., 1978.
13. Wilkes M.V., "Time-Sharing Computer Systems" (Second Ed.). Macdonald/American Elsevier Computer Monographs, N.Y., 1972.
14. Jones A.K., "Foundations of Secure Computation", p. 237-254, 1978.
15. Denning D.E., "A Lattice Model of Secure Information Flow. Comm. ACM 19, 5 (May 1976), 236-243.
16. Summers R.C. and Fernandez E.B., "Data Description for a Shared Data Base: Views, Integrity, and Authorization." Report G320-2671, IBM Scientific Center, Los Angeles, Calif., August 1975.
17. Krishnamurthy E.V., "Computer Cryptographic Techniques for Processing and Storage of Confidential Information," International Journal of Control, November, 1970.
18. Skatrud R.O., "A Consideration of the Application of Cryptographic Techniques to Data Processing, Proceedings of FJCC, 1969.
19. McFadyen J.M., "Systems Network Architecture: An Overview." IBM Systems. J. 15, 1 (1976), 4-23.
20. Green P.E., "An Introduction to Network Architectures and Protocols." IBM Systems J. 18, 2 (1979), 202-222.
21. Fernandez E.B., Summer R.C., and Wood

C., Database Security and Integrity, p. 267-285, 1981.

22. Senko M.E., "Conceptual Schemas, Abstract Data Structures, Enterprise Descriptions." Proc. Int. Computing Symp., Liege, Belgium, April 1977, 85-107.
23. Codd E.F., "Extending the Database Relational Model to Capture More Meaning. ACM TODS 4, 4(Dec. 1979), 397-434.
24. Smith J.M. and Smith D.C.P., "Database Abstractions: Aggregation and Generalization." ACM TODS 2, 2(June 1977), 105-133.
25. Zloof M., "Security and Integrity within the Query-by-Example Data Base Management Language." Report RC6982, IBM Research Laboratory, Yorktown Heights, N.Y., February 1978.
26. Eswaran K.P., "Specifications, Implementations, and Interactions of a Trigger Subsystem in an Integrated Database System." Report RJ1820, IBM Research Laboratory, San Jose, Calif., August 1976.
27. Gray J.N., "Notes on Data Base Operating Systems." In Operating Systems-An Advanced Course, R. Bayer et al., (Eds.), Springer-Verlag, 1978, 393-481.
28. Lindsay B.J. et al., "Notes on Distrituted Databases." IBM Research Report RJ2571, San Jose, Calif., July 1979.