Client/Server environment & Information Security

By

Ayesha Asghar

A thesis submitted to the department of Computer Science in partial fulfilment of the requirements for the degree of

Masters in Database Management

Lacrosse University
Department of Computer Science

March 2007

To my grandmother, Zainab
& mother, Khalida

Contents

ILLUSTRATIONS

Tables

PREFACE

Client/Server computing is currently one of the buzzwords in the computer industry. The client/server environment can be defined as an open systems environment. This openness of the client/server environment makes it a very popular environment to operate in. As information is exceedingly accessed in a client/server manner certain security issues arise.

In order to address this definite need for a secure client/server environment it is necessary to firstly define the client/server environment. This is accomplished through defining three possible ways to partition programs within the client/server environment.

Security or secure systems have a different meaning for different people. This thesis defines six attributes of information that should be maintained in order to have secure information. For certain environments some of these attributes may be unnecessary or of lesser importance.

Different security techniques and measures are discussed and classified in terms of the client/server partitions and the security attributes that are maintained by them. This is presented in the form of a matrix and provides an easy reference to decide on security measures in the client/server environment in order to protect a specific aspect of the information.

The importance of a security policy and more specifically the influence of the client/server environment on such a policy are discussed and it is demonstrated that the framework can assist in drawing up a security policy for a client/server environment.

This thesis furthermore defines an electronic document management system as a case study. It is shown that the client/server environment is a suitable environment for such a system. The security needs and problems are identified and classified in terms of the security attributes.

Solutions to the problems are discussed in order to provide a reasonably secure electronic document management system environment.

ACKNOWLEDGEMENTS

I would like to thank the auspicious instructors of Lacrosse University under whose guidance I’ve been able to come up with this thesis. I thank them for their cooperation through out and also for resolving all problems and every issue that I faced.

Next, I would like to thank my family who provided me with all the sources and material required for making this possible. Lastly; I would like to thank God Almighty who gave me the strength and will power to complete this piece of work.

ABBREVIATIONS

EDMS	Electronic Document Management System
OOP	Object Oriented Programming
GC	Garbage Collection

1. Introduction

1.1. Background Information

Throughout the world information is expanding at a phenomenal rate. In order to remain in business industry one has to keep up with this information explosion. This need for up to date information forces businesses to rely more and more upon the facilities provided to them by the computer industry.

Developments within the computer industry did not stagnate either. In fact, the information explosion and the technology explosion within the computer industry go hand in hand. The result of all this growth is that a large collection of different technologies exists today within the business community.

The need for these technologies to interact with one another brought many challenges to the computer industry. Research directions, such as client/server computing, federated databases and networking, paved the way for these different technologies to work together in an efficient manner.

These “new” technologies are leading to an increased level of data and other resource sharing between different users. This co-operative way of working has led to more openness in systems, in that data is more accessible and available. This rowing importance of openness in systems, places a definite burden on the security of the system. The notions “open systems” and “secure systems” seem to be contradicting one another.

The emphasis within open systems is to share information and other resources using a network. These networks were originally designed for ease of use and not mainly for security purposes.

The client/server architecture can be considered a specific case of an open system, in that the server provides the client with data and other services over a network. The next section will define terms typically used in the client/server environment.

1.2. Definition of Key Concepts

This section defines certain key concepts used throughout the thesis.

- ***Information security*** is the steps taken, tools implemented and mechanisms used to ensure that the hardware, software and netware are protected against incidental, intentional or unauthorized damage, harm or loss and that the hardware, software and netware are always available to the people that must get access to it [Eloff, 1994].

- ***Client/server computing*** divides the processing of tasks in an application between the network resources best suitable for the different tasks. These resources consist of two groups: clients and servers.

    1. ***A client*** is an intelligent workstation, typically a PC, with capabilities of doing local data manipulation and maintaining a user interface.

    2. ***A server*** is a computing device that is used with the specific intention of handling database functions or other numerical intensive processing [Sinha, 1992].

1.3. State of Art

The continued increase in the usage of computer systems to manage the information within organizations has also led to a bigger need for secure systems. The need for security and the need for more openness in a system can work detrimentally against each other. It is therefore necessary for companies to investigate which degree of security and which degree of openness should be allowed when developing secure open systems.

Currently companies world wide opt for “down-sizing” to a client/server environment, without top-management realizing all the implications thereof.

The network boom paved the way for the uprising of the client/server infrastructure [Strauss, 1993].

If companies are questioned as to why they are changing to client/server architecture the reasons are multiple [Martin, 1994; Schulteis, 1994]. Reasons include the retaining of current investment, the scalability of the architecture and an increase in application performance.

For the above reasons client/server systems are gaining popularity by the day. However, frequently security is given little consideration when the decisions are made.

1.4. Problem Statement

This thesis attempts to address the following questions:

- How much has been done to address the issue of security within the client/server environment?
- What are the functional security components of the client/server environment? How is this influenced by the specific client/server implementation?
- What influence does the client/server architecture have on the design/development of secure applications?
- Research has lead to well-defined standards and policies for the traditional centralized (resource sharing) and distributed systems. What makes the client/server environment

different from the more traditional architectures? Which information security standards are available and how are they applicable to the client/server environment?

- The application, together with its user interface, provides a means of accessing the resources available in the environment. What security measures need to be taken on the application level? How does the user interface influence the security of the system?

- In which way should the specific characteristics of a client/server environment be reflected within the computer security policy of an organization? How can security and integrity rules be defined for a client/server environment?

Some of the aforementioned issues were addressed partially by other authors. [Hart, 1995] identified the idea of different client/server partitions, but did not relate the partitions to information security needs. Other authors [Francis, 1993; Dixon, 1996], on the other hand, realized the need for security in the client/server environment, but failed to distinguish a client/server environment from a more generalized distributed open systems environment. Several information security issues in the distributed open systems environment have been addressed by various authors, for example [Gollman, 1993; Harris, 1994; Kruys, 1991; Reitenspies, 1993], but they did not indicate how these techniques and methods are influenced by the manner in which the applications are distributed across the environment.

This study shows that a synergy between the client/server partition and its information security needs exists.

1.5. Overview of this study

Chapter 2 describes the information security attributes that will be used to evaluate the level of information security of a client/server environment. Client/server implementations can be distinctly grouped according to the components residing on the client and the server respectively. In *chapter 3* the different client/server partitions are identified and discussed.

Chapter 4 ties the concepts of client/server partitions and security attributes together by proposing a security framework for the client/server environment.

Chapters 5 – 7 subsequently study the potential sources for security methods and techniques. *Chapter 5* studies evaluation criteria and evaluates it as a source of security methods and techniques applicable in the client/server environment. Current application development frameworks for open distributed environments are investigated in *chapter 6*. *Chapter 7* studies the World Wide Web and techniques that are used in this environment.

Chapter 8 consolidates the methods and techniques identified through studying the evaluation criteria, the application development frameworks and the World Wide Web in chapters 5, 6 and 7. As a result, the consolidated client/server security framework is presented.

Chapter 9 identifies certain problems in electronic document management systems and uses the framework to propose possible solutions.

Chapter 10 presents a possible structure for an information security policy in a client/server environment. Thereafter the components of the policy are discussed with special reference to the security framework presented in Chapter 8.

Chapter 11 summarizes the findings of this dissertation and identifies further research areas.

A bibliography of the literature consulted is provided and a copy of an article presented at an international conference is attached as an appendix.

2. Information Security Attributes

Traditionally the purpose of information security for information security specialists is to preserve the three security attributes: confidentiality, integrity and availability [Parker, 1995]. However, Donn Parker [Parker, 1995] presents a further three attributes of secure information: utility, authenticity and possession. This chapter will define all six attributes within the context of this thesis.

2.1. Availability

Information availability can be described as the ability to immediately obtain the information required for the accomplishment of a specific purpose.

Two areas impact on the availability attribute. Firstly the information should be accessible, i.e. it should be reliably stored in an appropriate place. Secondly the means to actually access the information should exist, i.e. the necessary computer infrastructure should exist and be at the disposal of whoever requests the information.

Deleting files
Accidental overwriting of files
Disk crashes
Infrastructure problems, e.g. network unavailability
Natural disasters

Fig. 1. List of possible threats to availability of information

In figure 1, a list of possible threats to the availability of information is given. The deleting of files off the magnetic storage medium results in the loss of availability since it cannot be retrieved by the average user. Note however that this may be influenced by the operating system under which the file system was created. In certain instances it may not result in a permanent loss of the information, but merely present an inconvenience to the knowledgeable user. In the case of a disk crash the loss could be permanent if the necessary precautions have not been taken. Accidental overwriting of a file could occur when an existing file is replaced by another, e.g. copying a file to the same name, resulting in the original information becoming unavailable. If the network infrastructure becomes unavailable the information, although still intact at storage level, cannot be accessed by the users and is therefore unavailable. Similarly natural disasters may cause the abovementioned network unavailability, the breakdown of equipment or the destruction of the magnetic media, resulting in information becoming unavailable.

2.2. Utility

The utility attribute describes the usefulness of the information. In other words, whether the available information is fit for a specific purpose.

availability of information
user interface design
loss of cryptography key
misrepresentation of data

Fig. 2. List of possible threats to information utility

In Figure 2, the list of possible threats to the utility value of the information is given. The usefulness of the information depends on the availability of the information, since unavailable information cannot be useful in any way.

Furthermore the usefulness is greatly enhanced by the proper design of a user interface. The user interface design should form an integral part of the system's development lifecycle to ensure that the information presented is meaningful to the user. Although all relevant information may be presented to the user the actual manner in which this is done, may to a large extent influence the utility value of the information. For example, information presented in a massive table may prove useless to users unless studied for days, whereas the same information depicted on an appropriate graph can be used instantly.

2.3. Integrity

Integrity of the information points to the unimpaired soundness of the information.

storage errors
electrical/magnetical interference
database design
unauthorized changes

Fig. 3. List of possible threats to integrity of information

As displayed in the figure above, the list presents the several threats exist that endanger the integrity of information. Storage errors, like the malfunctioning of a disk drive, could lead to the loss of integrity of the data that is stored on it. Similarly writing to a bad sector on a disk could cause the written data to be unsound.

Other chances of corrupting the information are during transmission of the information where electrical or magnetical interference can corrupt the data being transmitted.

Both of the above issues have to do with physical integrity, whereas database design points to the logical integrity of the data. This could be increased considerably by enforcing referential integrity during the design process.

It is also noteworthy to mention that the integrity of data depends on the ability to only allow properly authenticated authorized users to change information in the database. If unauthorized changes are allowed then the integrity of the information in the system can be seriously damaged.

2.4. Authenticity

Information authenticity represents an authoritative quality, indicating that the information is worthy of acceptance or belief.

masquerading as somebody else
forging of messages
altering existing information

Fig. 4. List of possible threats to authenticity

Two important aspects of authenticity are

- The authenticity of communicating parties and
- The authenticity of content.

If a possible intruder can masquerade as somebody else then the authenticity of the communicating parties is threatened. If the messages between communication partners can be altered or forged then authenticity of content can be lost.

2.5. Confidentiality

Confidentiality of information indicates the state of being private. It implies that access to the information can only be gained by authorized users.

reading of information by unauthorized people
deduction of information

Fig. 5. List of possible threats to confidentiality of information

Possible threats to confidentiality are listed in figure 5. The issue of users being authorized to access the information is of the utmost importance for confidentiality of information. If an unauthorized user can read the information it cannot be considered confidential. Sometimes confidentiality can be threatened by the mere knowledge of the existence of certain information. For example, if a company is in the process of considering the rationalization of staff then knowledge of the existence of such documents by staff, which could possibly be influenced by the rationalization process, could be very disruptive to the organization.

2.6. Possession

The attribute of possession indicates the state of being in control of information.

stolen hardware
natural disasters
stolen magnetic media
copyright and trade secret violations

Fig. 6. List of possible threats to the possession of information

As listed in figure 6 several threats to the possession of information exist. If the hardware containing the information is stolen then possession of that information is influenced. Possession can be lost completely if the stolen hardware is never recovered or it could be lost temporarily until the hardware is found. Similar arguments can be made in the case of stolen magnetic media and natural disasters. Exclusive possession is threatened by copyright violations, i.e. if a copy of the information is stolen, but one or more copies are still in the possession of the owner.

2.7. Conclusion

This chapter defined the different security attributes that need to be protected to have secure information. Different organizations, however, have different security needs. These attributes therefore form a part of the presented framework in order to allow an organization to choose the attributes that are relevant to their situation.

The following chapter will identify the different partitions that exist in the client/server environment, explaining how an EDMS could be implemented for each of these partitions.

3. Client/Server Architecture

Popular literature has made "client/server" one of the leading industry buzzwords. Yet a definition of what the term means is still not agreed upon [Orfali, 1994].

This chapter defines the term client/server. Thereafter it studies possible ways in which a system can be partitioned within a client/server environment. These partitions are then presented in more detail by evaluating how an electronic document management system could be implemented for each of the partitions.

3.1. Client/server architecture

One fact that exists when defining the term client/server is that by virtue of the name it consists of at least the following two components: clients and servers. The first controversy, however, arises just after this fact has been established, namely what is the role of the client and what is the role of the server. In order to answer this question the basic components firstly have to be considered:

- Servers

  The server can be seen as a resource providing a service to other resources. For example, a computer providing services for database management.

- Clients

  The clients represent the "other" resources requesting a service from the server. For example, a workstation running a program requiring information from a database.

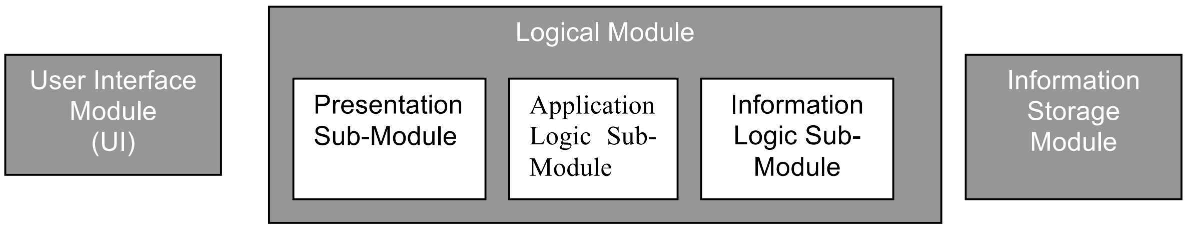

Logical Module
User Interface Module (UI)
Presentation Sub-Module
Application Logic Sub-Module
Information Logic Sub-Module
Information Storage Module

Fig. 7. Program partitions

A client/server system can be partitioned into clients and servers in several ways, similar to the ways that programs that run on a single machine are partitioned. Three modules are identified: the user interface module, the logic module and the information storage module (*Figure 7* refers). If all of the modules are situated on different machines then the arrangement is called the three-tier client/server model [Hart, 1995]. The logic module can be further subdivided into three sub-modules, namely the presentation logic sub-module, the application logic sub-module and the information logic sub-module.

The *user interface module* controls all user interaction, i.e. all input and output. The *presentation logic sub-module* processes the user input and prepares the user output. The *application logic sub-module* is where the computationally complex tasks are performed. The *information logic sub-module* determines which data is stored and retrieved and the manner in which it is done, e.g. it would determine the query which has to be sent to the database. The *information storage module* reads and writes data from the media where it is stored.

These modules can be distributed amongst clients and servers in several different ways within a client/server environment. If it is assumed that each of the five modules or sub-modules resides on maximum one machine and that they are distributed amongst at least two machines then 2500 (i.e. 4x54) different partitions are possible. Of course the assumption that each sub-module resides on maximum one machine is not necessarily valid since any specific sub-module could be distributed amongst many different machines. This results in an infinite number of possible partitions.

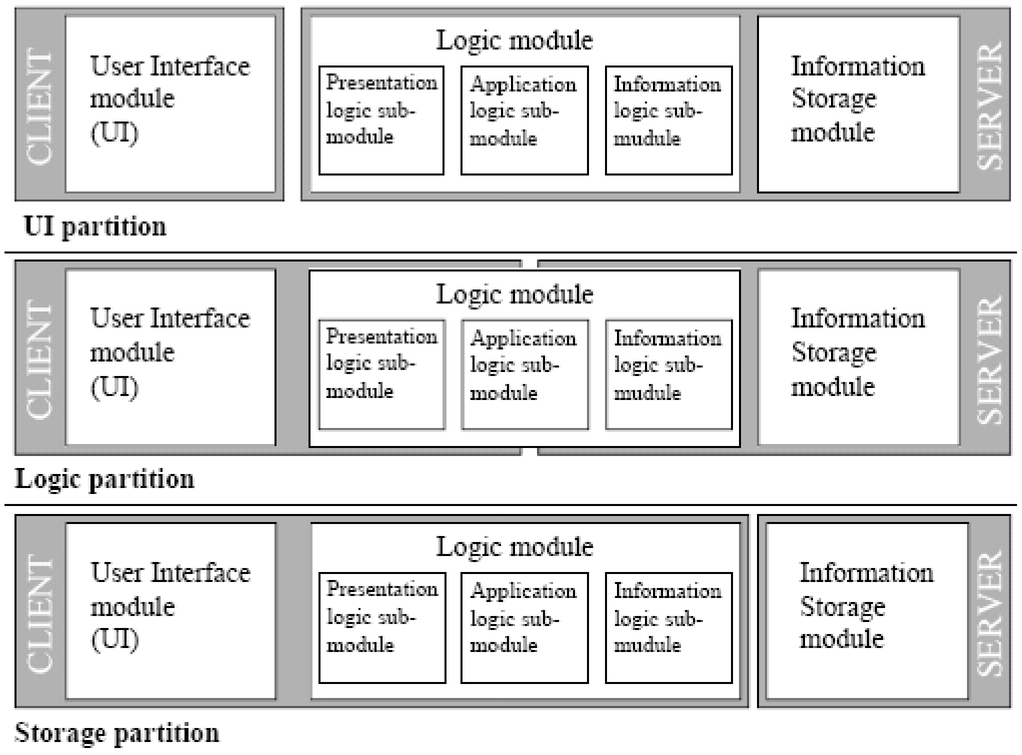

CLIENT
User Interface module (UI)
Logic module
Presentation logic sub-module
Application logic sub-module
Information logic sub-mudule
Information Storage module
SERVER
UI partition
Logic partition
Storage partition

Fig. 8. Three client/server partitions

For illustration purposes three possible partitions will be studied. These partitions currently are representative of the more popular approaches to client/server computing environments. These three partitions are depicted in Figure 8.

An UI partition is achieved by placing the user interface module on a different node (the client) than the logic and information storage modules (the server). Similarly a logic partition is achieved by grouping some of the logic modules and the user interface on one node (the client) and the information storage module together with some other logic modules on a different node (the server). A third possibility, a storage partition, is achieved by placing the information storage module on a different node (the server) than the logic and user interface modules (the client).

These three partitions, together with the security attributes discussed in Chapter 2, will form the basis for evaluating security in the client/server environment. To enhance understanding of the different partitions a more detailed investigation into each of these partitions follows.

3.2. The UI partition

This partition shows similarities to the typical mainframe environment. The server does all the processing and storage functions. The main difference is that the user interface module is located on the client, utilizing the graphical display capabilities of the client.

The information security requirements of the UI partition are similar to those of the more traditional environments in that the server is performing the identification, authentication and authorization functions. Either the database or the application, both residing on the server, controls integrity of the data.

The client workstation will only present the information. In other words, the client will present the information according to instructions received from the server. For example, the server will decide which document has to be displayed, as well as the manner in which the document should be displayed. The UI module, therefore, has extremely limited logic embedded in it, as it only has to know how to write to the output devices and how to receive input from the user.

To best understand this partition the execution of a query has to be considered. In this scenario it is assumed that a user wants to read *the memos that were sent by Mr. A.N. Other during September 1996*.

Figure 9 (next page) graphically represents this scenario:

1. The user requests "*the memos that were sent by Mr. A.N. Other during September 1996*" by using the input devices provided by the client workstation.

2. The user interface module on the client interprets the input and translates it into a format that is understandable by the presentation logic sub-module on the server. This could, for example, be “memos”;;;; “A.N.Other”;;;”September 1996”.

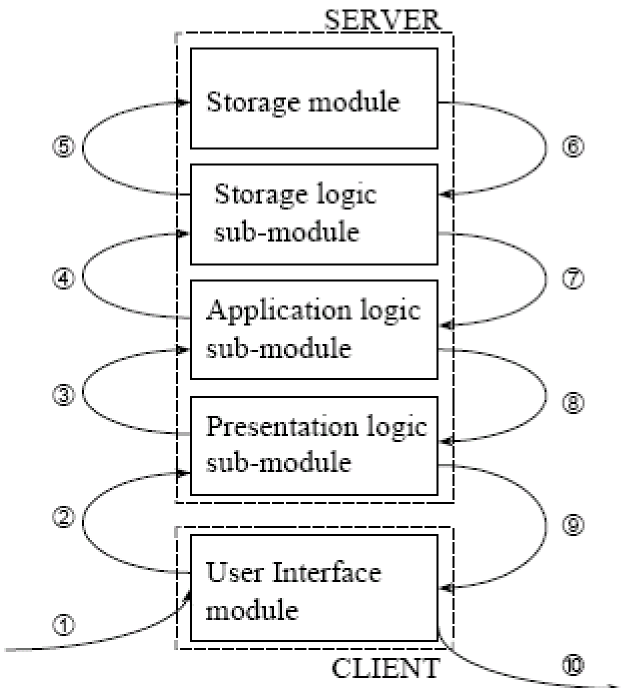


Fig. 9. The working of the different modules in the UI partition

3. The presentation logic sub-module interprets the input provided by the user interface module and passes it on to the application logic sub-module.

   For example:

   DOCUMENT_TYPE = “MEMOS”;
   AUTHOR = “A.N.OTHER”;
   ISSUE_DATE IN “SEPTEMBER 1996”

4. The application logic sub-module will further translate this application specific information in a way understandable by the information logic sub-module. This could, for example, be the following SQL-query:

```
SELECT DOCUMENT_NUMBER, FILENAME
FROM DOCUMENT D, PAGES P
WHERE D.DOCUMENT_NUMBER = P.DOCUMENT_NUMBER
AND DOCUMENT_TYPE = ‘MEMO’
AND AUTHOR = “A.N.OTHER”
AND TO_CHAR(ISSUE_DATE) LIKE ‘__-SEP-96’;
```

5. The information logic sub-module on receipt of the SQL-statement will do the logical translation to determine which data needs to be returned. It will formulate the necessary requests for the information storage module.

6. The information storage module translates the logical read requests into physical attributes, e.g. which disk, which sector and which blocks. It then executes the read requests, retrieving the required information. Thereafter the requested information is returned to the information logic sub-module.

7. The information logic sub-module receives the information and formats it in the way that it was requested by the application logic sub-module.

   For example:

```
13524 H:\dir1234\d1145873.wpd
13892 E:\dir9876\f8935467.wpd
14121 Z:\dir7845\e6428193.xls
```

8. The application logic sub-module does further operations on the information, if needed, and returns it to the presentation logic sub-module.

9. The presentation logic sub-module on the server decides in which manner to present the information to the user. In this example a tabular display of the information may be sufficient.

10. The user interface module on the client presents the information as requested by the presentation logic sub-module utilizing the output facilities of the client.

From the above explanation it is evident that the client has very limited responsibility in the user interface partition.

3.3. The storage partition

This partition represents a widely used client/server platform. Due to the fact that vendors frequently develop products for multiple platforms the application program often addresses certain issues, e.g. integrity checking. The main reason behind this is the different levels of offered security features between commercial databases. The server therefore only performs a storage function and all the other functions are shifted to the client.

Identification of the user is done by the client. Authentication and authorization (access control) involve both the client and server. Access to the application could be governed by the client, whilst access to the database is governed by the server.

Any application submitting information to the database must ensure that the integrity of the database is not damaged through manipulation of the values.

This however leaves the integrity of the database to “good faith” as any person who can gain access to the database could intentionally or unintentionally harm the integrity of the data.

In order to ascertain how a query will function within this partition the following task should be considered: *Store the index information of a new electrical drawing draughted by N.O.Green on 10 September 1996.*

Figure 10, below; depicts the execution of above task within a storage partition environment.

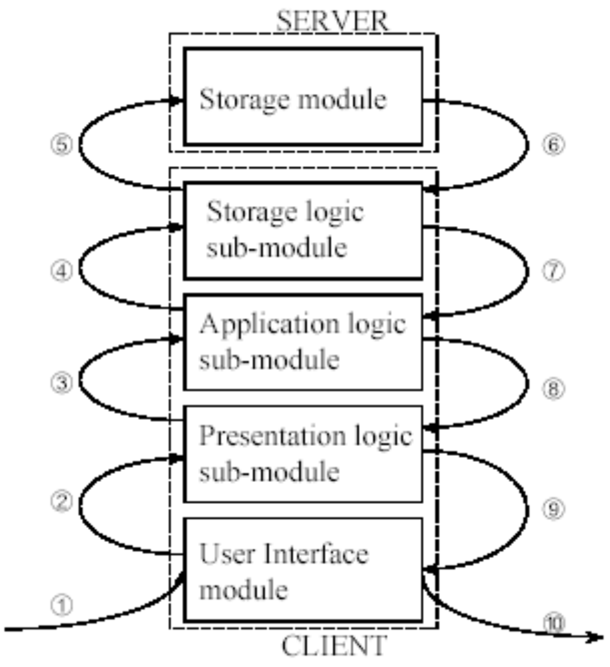


Fig. 10. The working of the components in the storage partition

1. The user issues a command utilizing the user interface presented by the UI module. In this case the command is "*to store the index information of a new electrical drawing draughted by N.O.Green on 10 September 1996*".

2. The user interface module interprets the input and translates it into a format that is understandable by the presentation logic sub-module.

   For example: "drawing";; "electrical";; "N.O.Green";;; "10 September 1996".

3. The presentation logic sub-module translates the request further, for example:

```
DOCUMENT_TYPE = "DRAWING";
DOCUMENT_DISCIPLINE = "ELECTRICAL";
AUTHOR = "N.O.Green";
ISSUE_DATE = "10 SEPTEMBER 1996"
```

The request is then passed on to the application logic sub-module.

4. The application logic sub-module interprets the input provided by the presentation logic sub-module. Firstly it performs several integrity checks. Amongst others this could include a check that certain values exist in other tables and that the issue date is not later than the present date. Thereafter a SQL-statement may be formulated to insert the necessary record into the database. This may include mandatory index fields, e.g. DOCUMENT_NUMBER, that possibly can be generated automatically. An example of such a statement is:

```
INSERT INTO DOCUMENT
(DOCUMENT_NUMBER,REVISION,DOCUMENT_TYPE,
DOCUMENT_DISCIPLINE,AUTHOR,ISSUE_DATE)
VALUES (“147856”,”01”,“DRAWING”,
“ELECTRICAL”,“N.O.Green”,“10-SEP-96”);
```

5. The information logic sub-module will logically translate the write request and pass it on to the information storage module.

6. The information storage module interprets the logical write request in order to perform the physical write. Once the physical write has taken place the information storage module returns a status code to the information logic sub-module.

7. The information logic sub-module interprets the status code and decides on an appropriate action. Actions could include the re-execution of the previous command and the return of a status code to the application logic sub-module.

8. The application logic sub-module’s behavior is governed by the value of the status code. If the status code implies an unsuccessful addition of the record then an appropriate corrective action could be decided upon by the application logic sub-module. A status code will be passed on to the presentation logic sub-module.

9. The presentation logic sub-module interprets the received status code and decides in which manner the information will be best presented to the user. In the case of a successful addition of the record a change to the status bar could be appropriate, whereas an unsuccessful addition of the record might necessitate the display of a message box with a descriptive error message and suggested corrective actions.

10. The UI module will receive the instruction to display the output which was decided upon by the presentation logic sub-module. Thereafter it performs the necessary actions to update the relevant output devices.

From the discussion of the storage partition it is evident that the client, for the storage partition, has a much larger responsibility than the server.

3.4. The logic partition

The ideal within a client/server environment is to utilize the client and the server for the functions that they are respectively good at. The client is best at displaying information and servers are best at storing information, which would imply that the UI module must reside on the client and the information storage module must reside on the server. The question that subsequently arises is "What about the logic module?" The logic module consists of three sub-modules that can be positioned in various ways. Since the client presents the user interface the presentation logic sub-module can be located on the client. The information logic sub-module can reside with the server as this is where the information is stored. The application logic sub-module could then reside on the client if no intensive processing is done; or it could reside on the server if much processing power is needed; or it could be split between the client and the server to appropriately share computing resources.

Consider the following scenario:

It is assumed that the user wishes to view document "ABCD". Document ABCD is a scanned image compressed using a highly efficient compression algorithm.

Figure 11 can be used as reference for discussion on how this will take place.

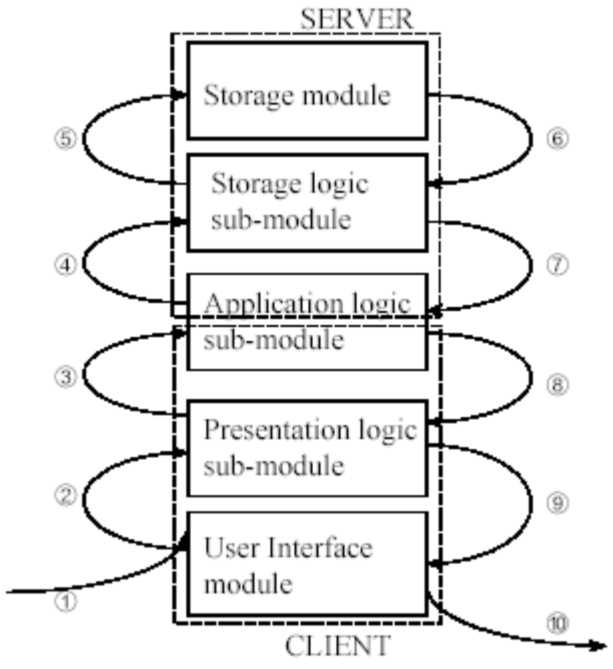


Fig. 11. EDMS and the logic partition

1. The user manipulates the user interface by using the various input devices to request to view document "ABCD".

2. The client interprets the input and provides the presentation logic sub-module with the corresponding request.

3. The presentation logic sub-module interprets the request and passes it on to the application logic sub-module. For example:

   ```
   REQUEST = 'VIEWDOC"; DOCUMENT_ID = "ABCD"
   ```

4. The application logic sub-module on the client evaluates the request and supplies the information logic sub-module on the server with the following request.

   ```
   SELECT DOCUMENT_IMAGE
   FROM DOCUMENT
   WHERE DOCUMENT_ID = "ABCD"
   ```

5. The information logic sub-module on the server interprets the request and translates it to physical read requests.

6. The information storage module returns the requested information to the information logic sub-module. The information logic sub-module formats the information as requested by the user, e.g. trims excess information off if the block that was read contains non-relevant information. The requested information is returned to the application logic sub-module on the server.

7. The application logic sub-module on the server does the interpretation of the information. In this case it will apply resource-intensive decompression techniques. The decompressed image will be passed on to the application logic sub-module on the client which may apply further operations on it.

8. The application logic sub-module on the client will pass the prepared image to the presentation logic sub-module on the client.

9. The presentation logic sub-module decides how the document should be viewed and instructs the UI module to display the document.

10. The UI module will utilize the output devices to communicate with the user.

This partition thus allows for effective balancing of loads. Clients perform generally less resource intensive duties, whilst servers execute the more resource intensive processes. This arrangement allows for the best properties of both the client and the server respectively to be exploited in order to obtain the most efficient application.

3.5. ORACLE and the various client/server partitions

ORACLE Corporation has a wide range of relational database management software available. It ranges from Personal ORACLE, for a PC, to ORACLE Enterprise Server, for a large distributed environment [ORA, 2007]. The next three sub-sections will discuss how ORACLE products can be used in the various client/server partitions.

3.5.1. ORACLE and the UI partition

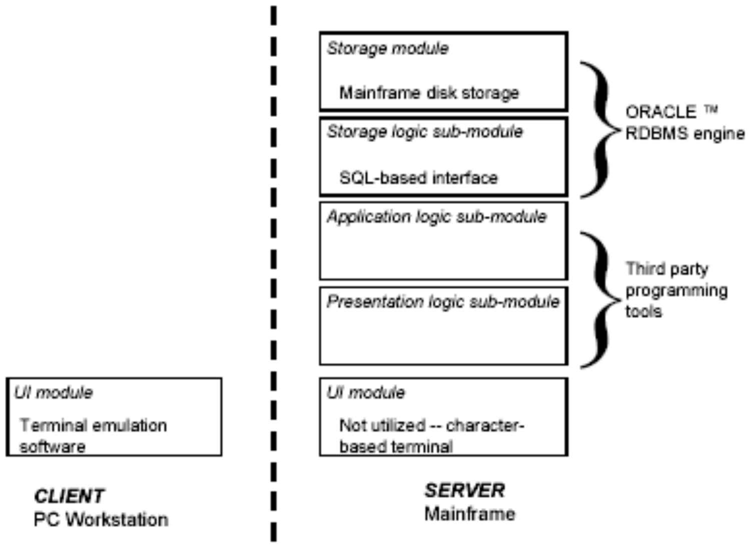

Storage module
Mainframe disk storage
Storage logic sub-module
SQL-based interface
Application logic sub-module
Presentation logic sub-module
UI module
Not utilized -- character-based terminal
ORACLE ™ RDBMS engine
Third party programming tools
UI module
Terminal emulation software
CLIENT
PC Workstation
SERVER
Mainframe

Fig. 12. Oracle and the UI partition

If an ORACLE database is used in the mainframe environment it is possible to do all the application development in the typical mainframe environment using character-based screens.

The UI partition can then be attained if the client workstations are set up to do some terminal emulation, possibly with some added features, e.g. mouse manipulation. The sole responsibility of the client then is to translate the user input into a format that is understandable by the presentation logic sub-module on the server. The aforementioned represents what was previously presented as the UI partition in Chapter 3. This scenario is depicted in Figure 12.

3.5.2. ORACLE and the storage partition

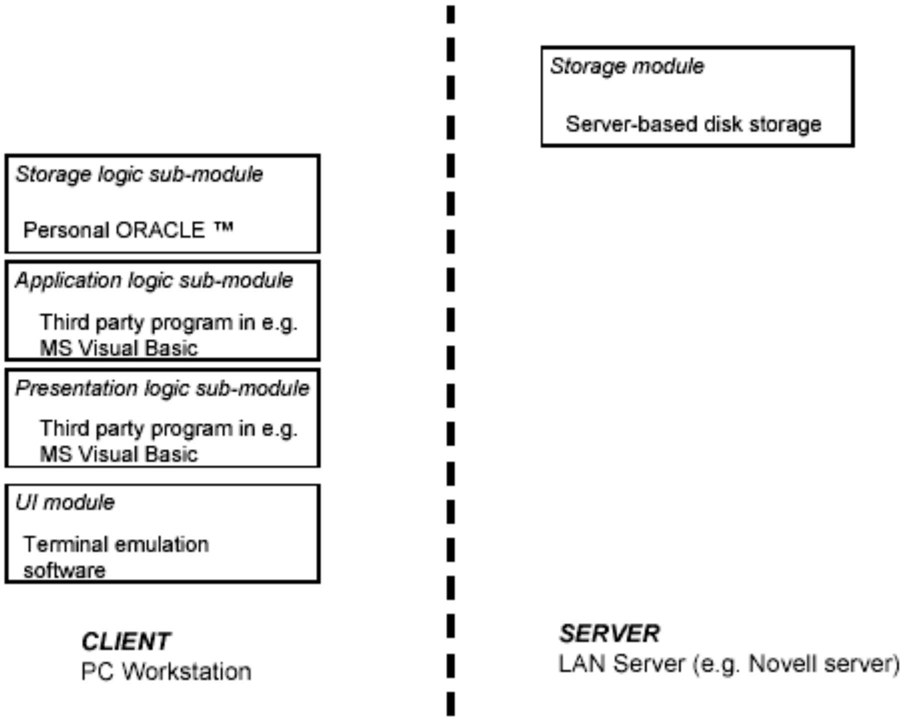

Storage module
Server-based disk storage
Storage logic sub-module
Personal ORACLE ™
Application logic sub-module
Third party program in e.g. MS Visual Basic
Presentation logic sub-module
Third party program in e.g. MS Visual Basic
UI module
Terminal emulation software
CLIENT
PC Workstation
SERVER
LAN Server (e.g. Novell server)

Fig. 13. Oracle and the Storage partition

Personal ORACLE [ORA, 2007] can be installed on personal computers in such a way that it can act as the information logic sub-module on the client workstation. The information storage module can still be set up to be on a server. Using Oracle's development tools or any third party tools that allow for communication with the database, for example Microsoft Visual Basic, application programs can be developed on the client.

The interface is presented using the windowing capabilities of, for example, Microsoft Windows. This situation is depicted in Figure 13.

3.5.3. ORACLE and the logic partition

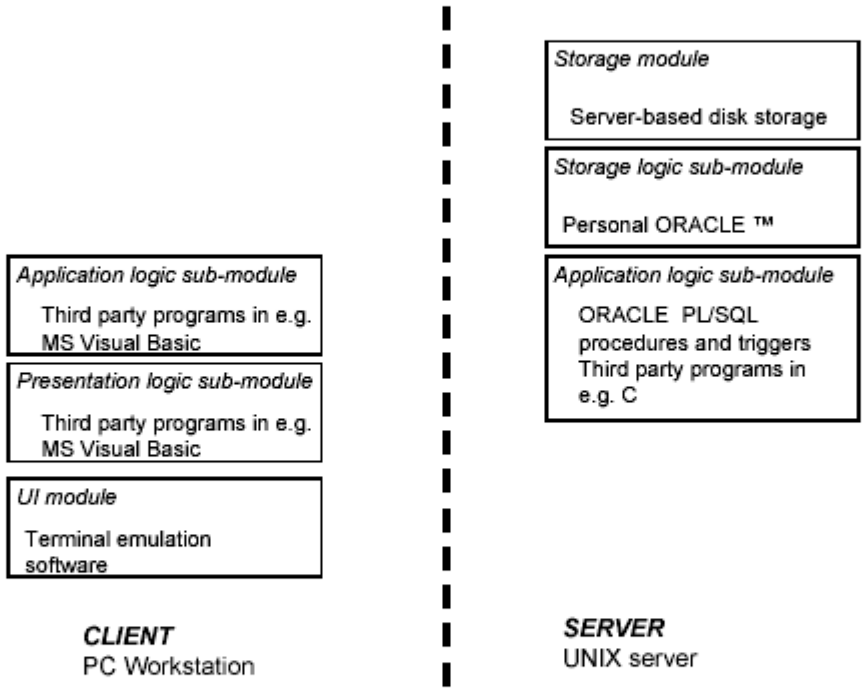

Storage module
Server-based disk storage
Storage logic sub-module
Personal ORACLE ™
Application logic sub-module
Third party programs in e.g. MS Visual Basic
Application logic sub-module
ORACLE PL/SQL procedures and triggers
Third party programs in e.g. C
Presentation logic sub-module
Third party programs in e.g. MS Visual Basic
UI module
Terminal emulation software
CLIENT
PC Workstation
SERVER
UNIX server

Fig. 14. Oracle and the Logic partition

The logic partition represents the most popular approach for installing ORACLE. The ORACLE database engine and storage will reside on the server. The user interface module is represented by the Windows capabilities.

The presentation logic sub-module can be developed using third party products on the client workstation. The application logic sub-module is represented on both the client and the server.

On the server this could be done using the PL/SQL language of ORACLE developing triggers and stored procedures [ORA, 1992]. Another possibility is to develop separate server programs which can act on special requests, using a language, e.g. C. In a UNIX environment this could, for example, be implemented as daemons, i.e. processes that are always running in the background.

3.6. Conclusion

This chapter presented three different client/server partitions for which the security measures that need to be implemented differ depending on the particular partition implemented and the required security features.

The next chapter will develop a framework based on the security attributes (chapter 2) and the client/server partitions (chapter 3) to be used when securing a client/server environment.

4. The Empty Framework

Chapter 2 defined the different security attributes that need to be preserved in a secure system. Various client/server partitions were defined and functionally discussed in Chapter 3.

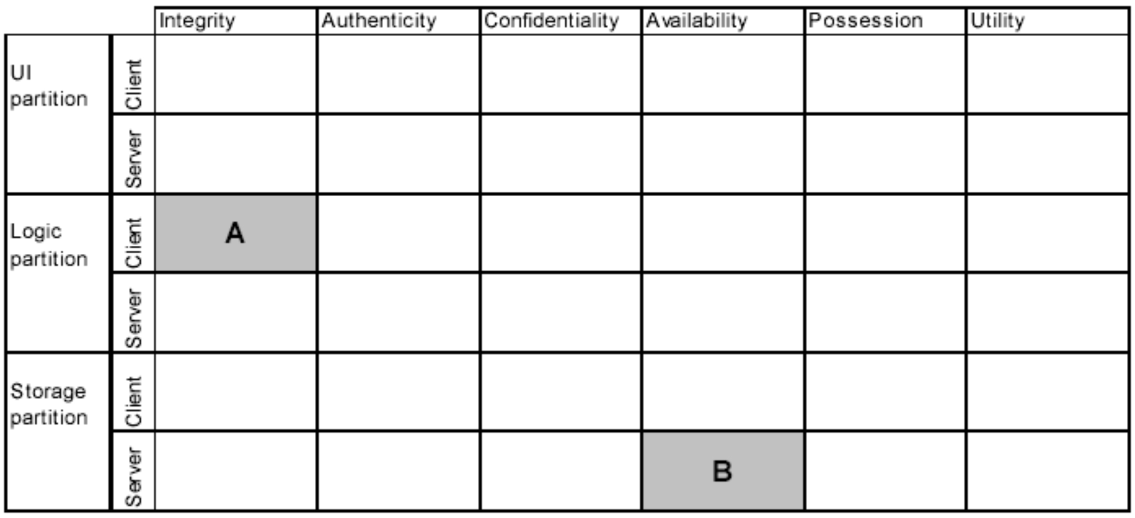
The questions “How can it be decided which security techniques are relevant when only some of these attributes require protection?” and “What influence does the choice of partition have on the security techniques that need to be implemented?” now arise. This chapter suggests a strategy for generating a framework that can be used to answer both the aforementioned questions.

4.1. The framework

The security techniques required may depend on the attributes that need protection as well as on the client/server partition implemented. The possible influence of the two aforementioned factors on the security techniques selected suggests that a two-dimensional matrix could be used to map the different security techniques.

Figure 15 presents the empty framework. The X-axis contains the six attributes of secure information: integrity, authenticity, confidentiality, availability, possession and utility. The Y-axis contains three client/server partitions: the UI partition, logic partition and storage partition. In addition each of these partitions is divided into a client and server part.

The highlighted entry “A” in Figure 15, therefore, presents the standards and techniques applicable to the client in a logic partition in order to preserve integrity of the information. Likewise the entry “B” presents the standards and techniques applicable to the server in a storage partition in order to preserve the availability of the information.



		Integrity	Authenticity	Confidentiality	Availability	Possession	Utility
UI partition	Client						
	Server						
Logic partition	Client	A					
	Server						
Storage partition	Client						
	Server				B		

Fig. 15. The Framework

4.2. 4.2 A strategy for completing the framework

This framework will be completed by exploring several issues. Currently several evaluation criteria exist. These criteria evaluate a system's security in terms of certain pre-set conditions that have to be met to qualify for a given classification. Chapter 5 will explore these conditions in order to determine their relevance in the client/server environment, as well as to determine at which security attributes it is aimed.

Within the open distributed systems arena certain application frameworks have been developed to set standards for developing secure applications for the client/server environment. Chapter 6 investigates these application frameworks and positions the accompanying techniques proposed within the framework.

An excellent example of client/server technology at work can be found in the World Wide Web (WWW). Chapter 7 contributes to the framework by studying security mechanisms applicable to WWW sites. Chapter 8 presents all of the findings of the previous chapters by providing a consolidated framework of security techniques.

5. Evaluation criteria and the Client/Server environment

Particular user communities drive most of the standardization efforts in the computer system security arena. The earlier work of the American Department of Defense (DoD) probably laid the groundwork for research in computer security [Reitenspies, 1993]. However, the continued growth in non-military and commercial type systems leads to the expansion of security requirements in those areas. A brief look at the Trusted Computer Security Evaluation Criteria (TCSEC), the Information Technology Security Evaluation Criteria (ITSEC) and the Common Criteria will be taken. This will be followed by a discussion of the role that the evaluation criteria will play, firstly in terms of the client/server environment in general and secondly in terms of specifically developing the contents of the framework.

5.1. TCSEC

The Trusted Computer Security Evaluation Criteria (TCSEC) of the United States Department of Defence (DoD) evaluates security according to evaluation classes D, C1, C2, B1, B2, B3 and A1 in order of increased functionality and assurance [DODS, 1985; O`Shea, 1991; Reitenspies, 1993].

Each class is identified by a set of criteria that addresses the security policy, accountability, assurance and documentation [DODS, 1985].

A distinction is made between the rating of products and the rating of installed systems. This is because specific systems, particularly when networked, may have certain vulnerabilities in spite of the evaluated ratings of the individual components of such a system.

5.1.1. TCSEC and the client/server environment

The most important aspect of the TCSEC with respect to the client/server environment is the differentiation between rating products and rating systems. In the client/server environment this is particularly useful as the installed system frequently consists of multi-vendor, multi-product installations.

The TCSEC can be used very successfully for evaluating certain components of the client/server environment. The TCSEC is particularly useful and strong in the evaluation of database security and operating systems. As pointed out in [CAEL92] the evaluation process is complex and very time consuming. The evaluation of a single component, e.g. the operating system, is a large task. The complexity of this task increases as the number of components increases. This is due to the complexity of the interaction between the different components. The TCSEC therefore seems a rather clumsy way of evaluating a complete system. [CAEL92] also points out that a commercial developer can at best perform an incomplete evaluation. This is mostly due to the fact that the developer needs a certain element of trust in the vendor, as it needs to be assumed that certain steps, for example design and development, have been fulfilled in the “proper secure way”.

Above scenario certainly holds in the client/server architecture where several products from potentially different vendors need to interact. Typically components such as the operating systems, network software, databases and applications will be sourced from different companies. An effective evaluation will then require co-operation from all the vendors involved, as well as an independent evaluation authority to facilitate the whole process. This “openness” may however necessitate the sharing of perceived “trade secrets”, which may cause certain vendors not to co-operate fully.

Bibliography

[Dixon, 1996]
Dixon, R., "Client/Server and Open Systems", John Wiley & Sons, 1996

[DODS, 1985]
Department of Defense Standard DOD 5200.28-STD, *"Department of Defense Trusted Computer System Evaluation Criteria"*, December 1985

[Eloff, 1994]
Eloff, M.M., *"Information Security in a Distributed environment",* M.Sc dissertation, RAU, 1994

[Francis, 1993]
Francis, R., *"The Search for Client/Server Security"*, Datamation Vol. 39, May 1993, pp. 39 – 43

[Gollman, 1993]
Gollman, D., Beth, T. and Damn, F., *"Authentication Services in distributed systems"*, Computers and Security, Vol. 12, pp. 753 -764

[Harris, 1994]
Harris, D. and Sidwell, D., "Distributed Database Security", Computers and Security, Vol. 12, 1994, pp. 547 - 557

[Hart, 1995]
Hart, J.M. and Rosenberg, B., *"Client/Server Computing for Technical Professionals"*, Addison-Wesley, 1995

[Kruys, 1991]
Kruys, J.P., *"Progress in Secure Distributed Systems"*, Computers and Security, Vol. 10, 1991, pp. 429 – 441

[Martin, 1994]
Martin RJ *"The Premise and the Promise"* Journal of Systems Management, January 1994, pp. 26 – 27

[ORA, 1992]
Oracle Corporation, "Oracle 7 Server Administrator's Guide", 1992.

[ORA, 2007]
Oracle Home Page, http://www.oracle.com

[Orfali, 1994]
Orfali, R., Harkey, D. and Edwards, J., *“Essential Client/Server Survival Guide”*, John Wiley & Sons, 1994

[O’Shea, 1991]
O’Shea, G.F.G., “Operating Systems Integrity”, Computers and Security, Vol 10, 1991, pp. 443 – 465

[Parker, 1995]
Parker, D., *“A new framework for Information Security to avoid Information Anarchy”*, Proceedings of the 11th IFIP/Sec ’95, May 1995, p.135 – 144.

[Reitenspies, 1993]
Reitenspies M *“Open Systems Security Standards”* Computers and Security, Vol. 12, 1993, pp. 341 – 361.

[Shulties, 1994]
Schulteis, R.A. & Bock, D.B., *“Benefits and Barriers to client/server Computing”,* Journal of Systems Management, February 1994, p. 12

[Sinha, 1992]
Sinha, A., *“Client-Server Computing”*, Communications of the ACM, Vol 35, No 7, 1992, p. 77-97.

[Strauss, 1993]
Strauss, P. *“LAN Boom paves the way for Client/Server”* Datamation, Nov 15, 1993, p.40-48