Fall 2008

An Investigation of the Security Designs of a Structured Query Language (Sql) Database and its Middleware Application and their Secure Implementation Within Thinclient Environments

Melissa D. Winner-Leoni
Regis University

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

An Investigation of the Security Designs of a Structured Query Language (SQL) Database and its Middleware Application and their Secure Implementation within Thin-client Environments.

by

Melissa D. Winner-Leoni
Winne411@regis.edu

A Thesis/Practicum Report submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Information Technology

School of Computer and Information Sciences
Regis University
Denver, Colorado

October 5, 2008

Abstract

The Information Portability and Accountability Act (HIPAA) and The Sarbanes-Oxley (SOX) regulations greatly influenced the health care industry regarding the means of securing financial and private data within information and technology. With the introduction of thin-client technologies into medical information systems (IS), data security and regulation compliancy becomes more problematic due to the exposure to the World Wide Web (WWW) and malicious activity. This author explores the best practices of the medical industry and information technology industry for securing electronic data within the thin-client environment at the three levels of architecture: the SQL database, its middleware application, and Web interface.

Designing security within the SQL database is not good enough as breaches can occur through unintended consequences during data access within the middleware application design and data exchange design over computer networks. For example, a "hospital's medical records, which are routinely exchanged over computer networks, are subject to the audit control an encryption requirements mandated for data security." (Department of, 2008).

While there is an overlapping of security techniques within each of the three layers of architectural security design, the use of 18 methodologies greatly enhances the ability to protect electronic information. Due to the variety of IS used within a medical facility, security conscientiousness, consistency of security design, excellent communication between designers, developers and system engineers, and the use of standardized security techniques within each of the three layers of architecture are required.

Acknowledgments

In the loving memory of Jacob, for without him I would have never learned the fundamental concept: *I am only limited by that which my mind cannot perceive.*

My heart filled thanks to all those in the medical industry who gave their time and information. I offer a special thank you to all the professors and advisors who worked diligently to assist me with my studies. A loving thanks to my greatest supporters who continually gave me encouragement, Esther, Elizabeth, Mary, Bill, and Hank. I could not have done this without them being a part of my life.

Table of Contents

List of Tables

Table of Figures

Executive Summary

According to A Guide to the Sarbanes-Oxley Act (2002) and the Consumer Bill of Rights and Responsibilities (2001), the implementation of laws in the United States (U.S.), such as Sarbanes-Oxley (SOX) and Health Insurance Portability and Accountability Act (HIPAA) has placed a burden on the medical industry to assure information privacy. This is an exploration of the methodologies for securing patients' personal, financial and confidential information contained within a medical SQL database, and securing its middleware application within thin-client environments by conducting interviews with Information Systems (IS) specialists within the medical community, and by evaluating IS documentation pertaining both directly and indirectly to the medical industry.

This document is written with a qualitative approach using grounded theory obtained from various resources including: documentation and information provided from IS specialists within the medical community, Oracle, International Business Machines (IBM), Institute of Electrical Electronic Engineers (IEEE), International Organizations for Standardization (ISO), National Institute of Standards (NIS), Association for Information Standards (AIS) and publications on technology security solutions. Other resources include medical organizations such as the Electronic Delivery of Information in Healthcare, medical journals such as the Health Industry Insights, and government agencies such as the U.S. Department of Health and Human Services (HHS).

This study provides an in-depth overview of database security design and its application security design and Web implementation design requirements. This study is relevant to readers who are responsible for implementing and maintaining security in

thin-client environments within not only the medical field, but also, more broadly across the Information Technology (IT) sector. The information provided is from standardization entities, and medical IT publications that compare and analyze the security designs of various medical SQL database and middleware applications within thin-client environments.

Due to a high variety of marketed security offerings by SQL database vendors, application providers, and thin-client providers, the issues involved in securing a medical database and its corresponding applications within a thin-client environment can vary significantly depending on the specific IS implemented (Bunker, 2007). As a result, no single security solution sufficiently meets every environment's requirements (Bunker, 2007). The purpose of this investigation is not to dictate what should be done, but rather to offer various solutions and methodologies at each level of architecture by focusing on three specific areas of concern: database-design specific security, database middleware design interface security and database/middleware-Web Server interface security design.

Chapter 1 – Introduction

The rules and regulations set forth by the Sarbanes-Oxley (SOX) and Health Insurance Portability and Accountability Act (HIPAA) has placed a burden on the medical industry to assure information privacy. From 1998 through 2001, those in the medical industry struggled with interpreting the laws and setting into place the data security that was necessary to meet their interpretation of the laws. The medical industry as well as many industries that were required to meet these laws did not have historical information which to build data security (History, Overview, 2008). Thus, many unique security solutions developed that gradually have in some part, become standardized (Bunker, 2007). With the introduction of thin-client technology, complex solutions are required. Information technology personnel have concerns regarding how to insure SOX and HIPAA security compliance between SQL databases, their n-tier applications, and Web interfaces. (Singhal, 2007)

.

1.1 – SOX Compliance

In Section 302 of the SOX Act (i.e. Appendix A), periodic statutory financial reports are to include certification of the following:

1. The signing officers are responsible for the internal controls and have evaluated these internal controls, within the previous ninety days, and have reported on their findings.
2. A required list of all deficiencies in the internal controls and information on any fraud that involves employees, who are involved with internal activities, is included with the financial statements.

3. Any significant changes in internal controls or related factors that could have a negative impact on the internal controls are included with the financial statements (*A Guide To*, 2002).

For the medical industry, these laws pertain to the organizations internal financial records including patient financial records and insurance carriers. A simple record within an SQL table of a medical billing database may contain data fields that require:

1. Limited or no access by certain personnel for viewing data records, which would require an "*As needs to know."* role within the organization.
2. Limited or no access by organization outsiders but related parties, such as: insurance providers, medical researchers and clinical trials personnel.
3. Access monitoring by activity log files noting users who have accessed a record, the date and time of access and changes made.
4. Audit table updates for each record when view, edits and updates occur, noting the user, the date and time the record was accessed

Table 1 reflects the role relationships in a simple SQL database table that you will note adds complexity to a secure database design. While there is some cross over between the SOX and HIPAA laws it is important to know that SOX is a regulation concerning internal financial reporting, presentation, and security while HIPAA is more representative of an individual's personal privacy.

1.2 – HIPAA Compliance

Chapter Six of the HIPAA rules and regulations (i.e. Appendix B) is on the consumer's privacy rights. Within the regulation, it states the health care consumer has the right to know that health care information is treated with confidentiality, that health

care information be protected and not shared with anyone, unless the consumer has provided written permission. There are three main requirements:

1. With very few exceptions, individually identifiable heath care information can be used without written consent for health purposes only, including the provision of heath care, payment for services, peer review, health promotion, disease management, and quality assurance.
2. In addition, disclosure of individually identifiable heath care information without written consent should be permitted in very limited circumstances where there is a clear legal basis for doing so. Such reasons include medical or health care research for which an institutional review board has determined anonymous records will not suffice, investigation of health care fraud, and public health reporting.
3. To the maximum feasible extent in all situations, non-identifiable health care information should be used, unless the individual has consented to the disclosure of individually identifiable information. When disclosure is required, no greater amount of information should be disclosed than is necessary to achieve the specific purpose of the disclosure (The Advisory Commission on Consumer Protection & Quality in the Health Care Industry, 1999).

As with the SOX regulations the above HIPAA regulations lends to thoughtful consideration to user roles for accessing information within a medical SQL database. Access to SQL databases and their tables are through key relationships to and from another table within one relational SQL database, or data mining across multiple medical databases it is important to know how each piece of data relates to the overall security of

information. Based on the SOX and HIPAA laws provided above, Table 1 is an interpretive example of how each field in a data record within one SQL medical billing table relates to each law.

Table 1 - Field Security by User Role

Field	SOX – Law	HIPAA - Law	User Activity Monitored	User Role	Requirement
Patient ID	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create Read Read Read Read
Last Name	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
First Name	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
Middle Name	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
Social Security Number	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update No Access Read, Update Read No Access
Street Address	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
Suite/Apt Number	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
City	No	Yes	Yes	Admissions Caregivers Billing	Create, Read, Update Read Read, Update

				Insurers Clinical & Research	Read No Access
Zip Code	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
Telephone No.	No	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update Read Read, Update Read No Access
Insurance Carrier	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update No Access Read, Update Read No Access
Insurance Group No.	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update No Access Read, Update Read No Access
Insurance Membership No.	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	Create, Read, Update No Access Read, Update Read No Access
Date Last Billed	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	No Access No Access Create, Read, Update Read No Access
Amount Due	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	No Access No Access Create, Read, Update Read No Access
Date Last Payment	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	No Access No Access Create, Read, Update Read No Access
Amount Last Paid	Yes	Yes	Yes	Admissions Caregivers Billing Insurers Clinical & Research	No Access No Access Create, Read, Update Read No Access
Status	Yes	Yes	No	Admissions Caregivers Billing Insurers Clinical & Research	Update No Access Create, Read, Update Read No Access
Last Accessed	Yes	Yes	Yes	Admissions Caregivers Billing	No Access No Access No Access

				Insurers Clinical & Research	No Access No Access

User and role access within a SQL database allow or restrict end users with views. The roles trigger a specific view that shows only the information the user or group is to see.

In review of Table 1, note that no user has delete functionality because under Medicare, the record retention period is 5 years and a patient can file a grievance on a HIPAA violation up to 6 years (Calloway, 2001/2006). Medical industry best practices state to retain all records, noting patient statuses such as, active, inactive, or deceased. Dependent on the institution or organization Information Systems (IS) personnel archive the patient information that exceeds the statute of limitations (D. Ricci, personal communication, August 8, 2008).

1.3 – SOX and HIPAA Impact

The impact of the SOX and HIPAA laws have on a SQL medical database, the application, and the Web interface require scrutiny of the types of data that require security. Mapping the data aids in identifying areas of concern and too, to note possible areas of conflicts. Exceptions to the SOX and HIPAA laws require some flexibility within the IS environment.

1.3.1 – Exceptions to the SOX and HIPAA laws

Under SOX, there are no exceptions to the laws that govern financial reporting, information security, and accountability. However, under the HIPAA laws there are several exceptions: medical or health care research for which an institutional review

board has determined anonymous records will not suffice, investigation of health care fraud, and public health reporting (The Advisory Commission on Consumer Protection & Quality in the Health Care Industry, 1999). Thus using user role controls within the database may become convoluted, impede a user from accessing needed information, and role changes made after the fact, within the IS may possibly break the chain the security throughout the database, application and Web interface. This is not to say that user roles should not be used for security purposes, it is however, important that user roles are planned and designed throughout the IS, and used with other security options to allow for exceptions and user flexibility.

1.3.2 – Flexible Security Solution with SOX and HIPAA Compliance

The SOX and HIPAA regulations are to protect those with invested interest and the consumers. While the laws have made it clear what must be done they do not define how it is to be done. Not knowing how, has made it difficult for those who are responsible to insure the laws are upheld and has especially been difficult for those working within Information Systems (History, Overview,, 2008) While there are many IS security solutions, not all are viable solutions for every organization (Bunker, 2007). Much planning is needed for security processes through database design, n-tier application development and Web implementation (Singhal, 2007) so as not to break the chain of security at different levels of technology, yet not burden the end user with remembering a multitude of passwords, or impeding their ability to work efficiently.

Chapter 2 – Review Literature and Research

In McGovern's book *A Practical Guide to Enterprise Architecture,* McGovern defines thin-client technology as accessing a network either by Internet browser or specialize client software, to processes information where the client software primarily focuses on conveying input and output, not processing data. Thin-client is the direct opposite of fat-client technology where the majority of processing takes places on the user's computer and client software and the server is a main storage device. In the thin-client environment, significant data processing occurs at server (McGovern et al., 2004).

Breaches of security through knowledge of an approved user credentials, blended threats, SQL injection and other malicious activities do occur. In an article on data security, Litwin states the most vulnerable points that a breach can occur are at the internal user and workstation, the remote user and workstation, and the Internet. Breaching of the network hardware security mean data protection is dependent on measures incorporated into the databases, the applications and the Web interface (Litwin, 2008).

Figure 1 depicts a typical networks security configuration for thin client technologies. While hardware configurations and third party security solutions are not part of this study, a quick overview of the configuration is helpful in understanding thin-client technologies. The uses of firewalls, secure switches, and address masking aid in achieving maximum hardware security. In some facilities, an additional internal firewall, often referred to as the *Clean Firewall* is present. The Clean Firewall monitors all internal Web network activity. The firewall external to the network is the *Dirty Firewall*

and monitors both incoming and out going Web activity. (B. Williams, personal communication, July 28, 2008).

2.1– **Networking Security**

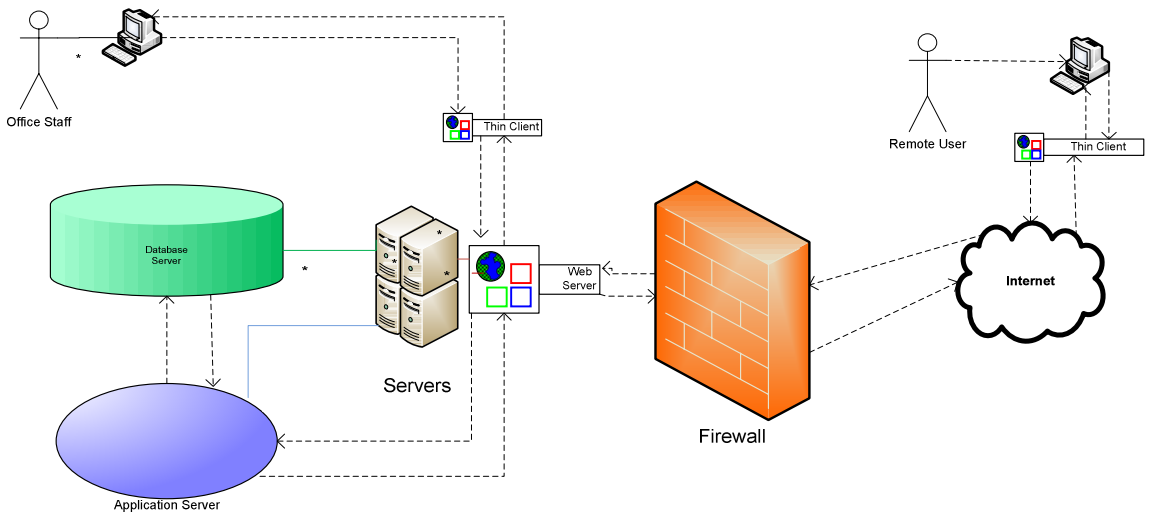

Figure 1 - Typical Network Configuration

Meier in an article on data security for Microsoft's Web products presents a visual representation of known Web server vulnerabilities shown in Figure 2. These vulnerabilities are internal and external to the network. In a medical organization, any one of these breaches will place sensitive patient and financial information at risk.

1. Information disclosure and code disclosure obtained from reading unsecured source code on the Web Site risks unauthorized application changes and malicious alterations to IS systems.

2. Arbitrary code execution in unrestricted data field entry, results in malicious cross-site scripting, SQL injection, and path transversal, that result in unauthorized access to data and information and compromised data.
3. Profiling, is port scans, ping sweeps, banner grabbing and NetBIOS-Enumerations that interfere with authorized user access, responding with information for unauthorized user access, and malicious interference of IS operations.
4. Denial of Service causing buffer overflows and SYN Floods, that directly blocking authorize user access.
5. Viruses, Worms and Trojan Horses that are malicious code and programs that directly compromises data and information, and replicate throughout the IS systems.

Programmed to monitor and thwart these types of vulnerabilities the *dirty* firewall is the first line of defense externally to the Web Server. The internal and *clean* firewall is the second line of defense between the Web Server application and the database. Once a breach of security occurs at the Web server, it is up to the security designs of the applications and databases to protect data to meet SOX and HIPAA requirements for privacy of information.

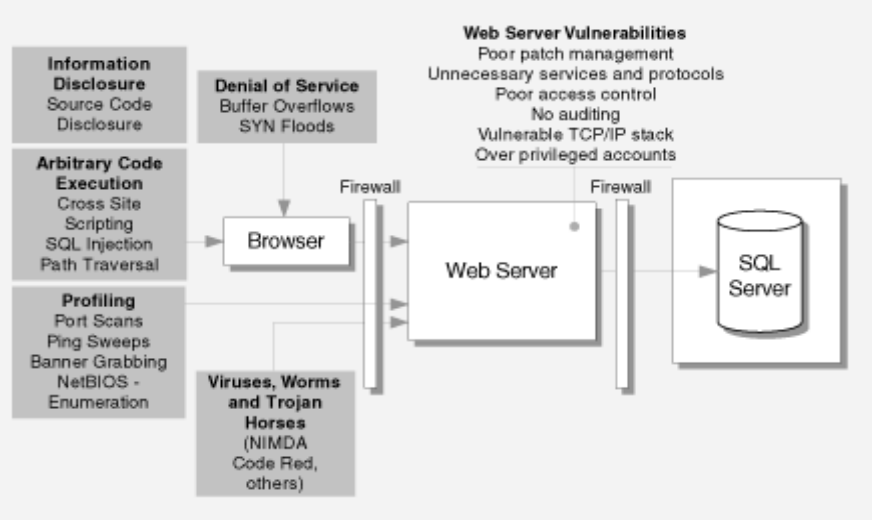

Figure 2 - Web Server Vulnerabilities

(Meier et al., 2003)

IS personnel incorporate the use of user identification (user ID) and password combination as the first layer of network access security. According to Singer, IS personnel often configure user ID and password combinations to replicate throughout the network to allow for ease of user access to various databases and applications. By replication of user ID and password, end users do not have to reenter their identification for each database or application on the network, thus making user access easier and expedient. However, not every user needs access to every database and application on the network, therefore each database and application needs added security measures (H. Singer, personal communication, July 28, 2008)

The user ID and password combination allow a user access to a network, then once on the network, access to applications or access to database. Each user ID assigned to a role determines what applications and databases the end user has access. Roles are assigned to Groups, thus for a user whose job function is admitting patients who come into the emergency room (ER), the user ID is assigned to the role admission within group ER.

2.1 - User ID and Passwords

The baseline of user security begins with whom to allow access to a database, an application, the Web interface and the network. The Institute of Electrical and Electronics Engineers (IEEE), National Institute of Standards and Technology (NIST), and other standardization groups provide standardizations for setup and design, yet each organization must decide the impact of their added security design on the users.

According to the standards set by the Office of Information Technology Solutions (OITS), IEEE, and the NIST, there are four standardizations of user ID. One states user first letter of first name and a minimum of seven characters of the last name. If the name has fewer than seven characters, or is a common name that has several users such as John Smith, then numeric assignment to fill the last digits, thus jsmith becomes jsmith01, jsmith02. The second standardization is first name.last name with the same numeric fill as above for common names, john.smith or john.smith001. The third is first name _last name with the same numeric fill as above john_smith or john_smith001. A fourth standardization is the last name only with a numeric fill as above, for example smith001, smith002. A standard is eight or more characters, dependent on the operating system or the use of legacy systems that are eight-character restricted.

The password standardization is no fewer then eight characters that are case sensitive with at least one uppercase, one symbol, and one numeric value (IEEE, 2006). An example is the password *gandolf* could then be *6@Nd0lF*.

In addition, there are other security methodologies used rather than sole dependency on ID and password access to secure data and information such as:

1. Data encryption within the SQL database or the application based on the user ID and role.
2. Based on the user's role limited views of data and information.
3. Based on the user's role limited menu access to an application.
4. Based on the user, sessions limited to access during known working hours.
5. Use of audit logs on network access, and database or application or both that records, Creation, Read, Update, and Deletion (CRUD) activity along with the user ID, date, and time of the event.
6. SQL audit table inserts of records field update activity along with the user ID, data, and time of the event.
7. SQL audit table inserts of the use of Dynamic Linked Libraries (DLL) that trigger table drops or reconfiguration.
8. SQL updates that insert the last user ID and the date and time of CRUD activity to the record.
9. Pseudonymization, a technique where identification data is transformed, and replaced by a specifier that cannot be associated with the identification data without knowing a certain secret
10. Use of certificates, Secured Socket Layers (SSL) or Kerberos to allow a user access to a network, database, or application based on need, role or group.

According to McGovern, any combination of the above are used to secure databases, applications and Web interface. The more combinations that are used, the more complicated security becomes. Reviewing and testing security practices for their effectiveness is a good practice. While strengthening security is important, there is a

chance of interfering with a user's ability to work efficiently within their software environment (McGovern et al., 2004).

2.2 – Database Security

Oracle Corporation, Microsoft Corporation, Novell, and International Business Machines Corporation (IBM) to name just a few, have incorporated the use of user roles into their network, database, and application management. The premise of roles is to separate users into groups that allow or denied access to specific software modules or database tables based on job requirements and what information the position needs to know. Yet, this is one of the first critical areas for a breach of security between the application and the database, as shown in Figure 3 (H. Singer, 2008).

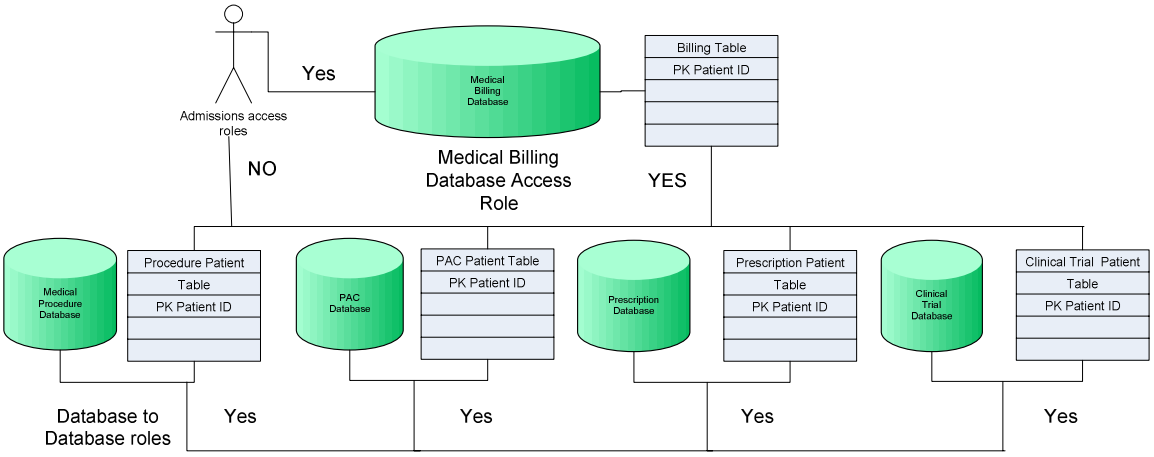

Figure 3 - User and Database Roles

While the admissions personnel have access to the medical billing patient table within the medical billing database, they do not have a need to access to any other database on the network. Yet, the medical billing database itself does have permissions to access other databases on the network through the primary key field that is the patient

ID. A breakdown of security occurs when the user role access changes from the originating user role to the systems database role. The systems database role normally has administrator privileges, allowing it free access to perform actions that the general user cannot such as, accessing a trigger that will update tables that the user is not aware of but are important to data integrity. Thus, when admissions personnel seek additional information on the patient ID, the other database(s) receive a systems request from the medical billing database and allows for the CRUD activity of the patient information. According to Barak in his article, *The Proactive Security Toolkit and Applications*, this misinterpretation of the user role can occur within the database security design or the application security design (Barak, Herzberg, Naor, & Shai, 2002).

Another related security breach can occur when a SQL savvy end user knows the names of SQL tables that they do not have user access permissions. If this user has the ability to create tables on the fly within a database, the user may then have the ability to extrapolate data from a restricted table into the users newly created and owned table without recognition by the database as a denied user. This user may also have the ability to assign other users access rights to the newly created table that then offers those users the ability to view data that they would not normally have access or be privy to. According to Greenwald in his book, *Oracle Essentials,* this occurs when the request for the data to the protected table looks as though it is coming from the system, or the system administrator and not the originating user (Greenwald, Stackowiak, & Stern, 1999/2004).

2.2.1 - User Roles and Groups

Dependent on the organization there could be many roles for groups of users. Such as role by job function and grouped by departments are used. In a hospital

environment, this could be admissions – radiology, admissions-pediatrics, and admissions-cancer-center. It may not be necessary for the admissions personnel within one healthcare specialization to see the patient's private information within another area specialization, therefore having the personnel grouped by specialization also adds security (D. Ricci, 2008). Database and application developers design views, encrypt fields, disable or hide menu offerings, and control general access with the user roles and groups in mind (Watson et al., 2005).

2.2.2 - SQL Database Views

According to Christensen in his book, *The Project Manager's Guide to Software Engineering's Best Practices,* there are no hard rules or standards presented by any of the standardization groups that state all views should be created in the database rather than the application presentation layer, or conversely (Christensen, Schneider, Thayer, & Winters, 2001/2003). There are reasons one would be the choice over the other.

1. If creating a new SQL database placing the views within the database is a good security choice. The reasoning being, is the application developer can then script to a procedure call based on the user role to the views, which then adds a layer of security.
2. If using an old SQL database that contains usable views then placing all the views within the database makes the same security sense as in item 1 above, but also preserves design continuity.

3. If creating an application that crosses platforms with legacy SQL databases and nonSQL databases then it makes sense to develop the views within the presentation layer of the application based on the user roles
4. If creating an application for front-end security so as to hide the database, creating views with alias naming conventions (discussed later under applications) is the choice (Christensen, Schneider, Thayer, & Winters, 2001/2003).

While restricting views so an end user is not aware that certain information exists is an excellent security measure, it may not always be possible. Data and information pulled from legacy system may require data encryption based on user groups and roles.

2.2.3 - Data Encryption within the Database

The encryption method to use is dependent on the SQL database and the product version. At this writing, the most recent versions of Microsoft SQL, Oracle, and Sybase have Advance Encryption Standard (AES) encryption capability. Earlier versions of these products offered Data Encryption Standard (DES) encryption except for Microsoft Windows 2000, which did not allow for in record data encryption (Andrews, 2001).

The National Institute of Systems Technology highly supported DES that applies a 56-bit key to each 64-bit block of data. That offered 72 quadrillion or more possible encryption keys. However, in October of 2000 the NIST opted to drop their standard of triple-DES for the use of AES as it comes in 128, 192 and 256-bit versions (Smid, 2001). When working with data field encryption, developers must plan and be aware of the ramifications, as it has occurred that a developer has locked everyone, including himself from viewing data (Woody, 2007). Encryption will slow the retrieval of data since the

data will have to decrypt for permitted user roles to view the data. For imaging SQL medical databases such as Picture Archive and Communication Systems (PACS), encryption may not be a practical option (Riedl, 2008). Pseudonymization may offer an alternative solution.

2.2.4 - Pseudonymization

In his article for the IEEE, *Pseudonymization for Improving the Privacy in e-Health Applications,* Riedl defines Pseudonymization as a technique where transformation of identification data occurs and replaced by a specifier. The specifier cannot be associated with the identification data without knowing a certain secret (Riedl, 2008) such as a Personal Identification Number (PIN). This technique is encryption for the entire database, not a field or a table, and the database must have at lease two tables. Riedl states, that one of the tables is that of protected information, and the second table being the table that contains the pseudonyms and the pseudonymized data. Known as depersonalization, the process is to remove direct association between the personal information and the person's identity. There are two possible ways of depersonalization, one is with the use of algorithms for calculating the pseudonym and the second is hashing techniques. Hashing techniques are the least secure, as reversal of hashing requires access to a stored list of all the pseudonyms. A breach of security can compromise the stored list and then the data (Riedl, 2008).

Riedl, states that encryption provides a more secure alternative for building pseudonyms. The process uses a symmetric algorithm to create a secret key that paired with a user's asymmetric key allowing access. Because these keys must be kept secret, they are best applied to smart cards systems or key fob systems as this eliminates

performing cryptographic operations on open systems, hence one more layer of security. The down side is that the cards and fobs can be lost or stolen (Riedl, 2008)

2.2.5– SQL Database Update on CRUD

A practice of some medical organizations is to update the SQL database record each time a record is touch (T. DeLuca, personal communication, August 2, 2008). The record update takes place as the record closes, and an update to a *last access* field with user ID and a time and date stamp. While considered a good practice within the medical industry, to note the user who was the last to create, read, or update a record, there lacks the history of every user who has touched the record. Ricci stated that this practice is good for SQL queries for searching database tables to monitor user activity and the possible attempt or actual breach of security (D. Ricci, 2008).

2.2.6 - Audit Logs and Audit Tables

According to Swanson in her document for NIST standardization, *Generally Accepted Principles and Practices for Securing Information Technology Systems*, audit logs are at the server level, the application level and the database level. Used to monitor user activity at each level, audit logs are useful in determining if someone has attempted or has breach security. The server level, audit logs record each user's access, the time of access and the time of user exit. The application and database developers determine the activity recorded within the application and database audit logs. The NIST recommend audit logs in both the database and application (Swanson & Guttman, 1996/2008). The logs are plain text updates that record user SQL, CRUD, and operational activity which

can be read and therefore compromised. Writing audit logs to obscure disk locations and not necessarily on the same server add an additional layer of security. Audit logs offer accountability, activity reconstruction, intrusion detection and problem detection information (Swanson & Guttman, 1996/2008).

Padilla in his article written for the IEEE, *Don't Gamble with HIPAA Security Compliance*, states that tracking of user updates to specific data fields and inserting the information into an audit table is another means of meeting SOX and HIPAA security requirements. The premise of audit tables is a trigger that will activate on a SQL *UPDATE* command that will update an audit table with the old information, the new information, the user ID, and date and time stamp. Not all record field activity requires this type of audit however, when items such as, patient name, social security number, insurance carrier, employer, disease diagnoses, pharmaceuticals, and other protected information are altered, a record of who made the change, the original data, and the changed data is a good practice (Padilla, 2005).

2.3 – Application Security

In his book *Object-Oriented & Classical Software Engineering*, Stephen Schach states that Object-Oriented Programming (OOP) is quickly becoming the industry standard for application development. OOP is the foundation of Web based applications and is layered n-tier (Schach, 2007).

In his book *Visual C# 2005*, Watson states, the theory of OOP is to keep application coding reusable, consistent, and expedient. The coding patterns make it easy to change one class without affecting another class or taking a program off-line (Watson et al., 2005). A design flaw or a change in code may cause a breach within the security

design. As discussed in a previous section an undesirable effect would be for a users' role to change to a systems role when requesting data or information.

An example of role changes within is the application software is the admissions personnel in the ER group create a new patient record (Presentation Layer). The ER-Admissions role has limited access to the billing database (Data Storage Layer). However, a business rule (Business Logic Layer) states *to avoid the possibility of duplicate patient information, a newly created patient ID must replicate to all patient databases throughout the organization*. The ER admissions personnel do not have access to all of the other databases (Data Storage Layer). The business logic layer calls a procedure within the database or application, triggering an event with administrative permissions to update all patient relational databases with the new patient ID. This event occurs in background however, the user's permissions should remain in place for all subsequent transactions.

Each layer of the n-tier design, an example of which shown in Figure 4, is for purpose and functionality of each application. Layered n-tier design applications can either attach to a SQL database or written to contain data within an application data file. The code containing scripts to attach to a database or output to a named file is located in the data layer of the n-tier application. There is also a layered n-tier known as the Model View Controller (MVC) that directly interfaces with a SQL database from the presentation layer. While n-tier layer design looks exactly like MVC design, the major difference is that the MVC allows direct interface with the database from the presentation layer (McGovern et al., 2004).

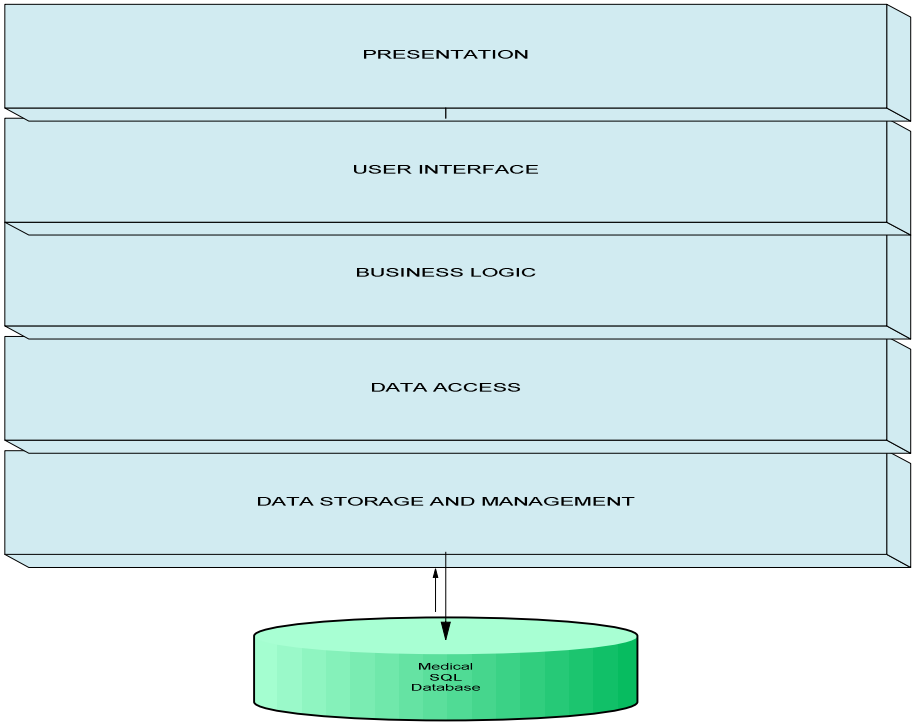

Figure 4 - N-tier design layers

According to Monroe in his book, *Architectural Styles, Design Patterns, and Objects: Software Engineering,* many of the security method, such as, user views, data restriction, and data encryption, that are available within SQL database design, are also available within application design. There are no hard rules regarding which security method the application handles, and those that the database handles (Monroe, Kompanek, Melton, & Garland, 2003). Not all layers of the n-tier design are specific to security or should have a security purpose.

2.3.1 - Data Storage and Management

There are two schools of thoughts on the data storage and management layer. The first is allowing access over a network and the second is placing the database behind a set of database handler routines. According to Britton in his book *IT Architectures and Middleware*, allowing access over a network exposes the database to a malicious user.

Hiding the database behind database handler routines is old technology but allows the database design to change while preserving the original interface (Britton & Bye, 2004). While the MVC model does permit OOP applications direct linking to the SQL database, OOP allows for the use of data field aliases which aides in shielding the database tables and their data fields. According to Britton, the preferred method is adding an additional layer of security by hiding the database behind a set of database handler routines.

2.3.2 - Data Access Layer

The data access layer provides a simplified means of accessing the data. Rather than using the standard means of an SQL statement such as *Update* and *Delete*, procedure names that call SQL coded within the database are used, thus allowing the database to manipulate the data. According to McGovern, using database procedures is an excellent means of cloaking the database operations from view and from those who do not need know how the database works. Within many organization's it is common for the middleware application developers to have the knowledge of what procedure to call, yet not the intricate knowledge of how the procedure works within the database (McGovern et al., 2004). Singer states that while allowing the application developers access to procedures and not having the complete knowledge of the database is going to extremes to protect information, HIPAA does state that non- identifiable health care information should be used to maximize security (Singer, 2008).

2.3.3 - Business Layer

Christensen in his book *The Project Manager's Guide to Software Engineering's Best Practices,* states that the business layer compartmentalizes calculations and the

business rules. Also divided by activities, the code within the business layer is responsible for pulling a medical image from a PACS database or a recent prescription from a pharmacy database based on user request. As stated earlier, the risk is the code call for data access using administrative permissions to a secondary database for one purpose and not returning the original user permissions thus, giving the user access to data that the user should not have access to (Christensen & Thayer, 2001).

2.3.4 - User Interface Layer

Britton in his book, *IT Architectures and Middleware,* states the user interface layer is the directions to the business layer to respond in a certain way when the user enters information, clicks on a button, or depresses key within the presentation layer view. The user interface layer is solely for functionality within the presentation layer and workflow and not for security (Britton & Bye, 2004).

2.3.5 - Presentation Layer

The presentation layer is what the user see's. If not using the SQL database views, then the views based on the user role, are part of this layer, as are active menu selections. According to Britton, if the application requirements call for the use of encryption, this is the layer where the encryption scripting takes place (Britton & Bye, 2004). As with database encryption, the NIST recommends AES.

2.4 - Web Publishing Security

Publishing an application to a Web server varies on the software and technology used. Each supplier has security tools to aid in publishing and securing a Web Portal using standard security tools:

1. Use of Secure Sockets Layer (SSL) that is the IT Industry standard for security technology that establishes an encrypted link between a Web Server and a users browser using Public Key and Private Key certificates.
2. Public and private keys certificates also work with a third party entity known as the certificate authority to match the keys. Private keys are created with several lines of data known only to the host, are digitally encrypted using a mathematical formula. The certificate authority validates the user and the public key, and then matches the public key with the private key to allow access.
3. Kerberos is an authentication protocol developed by Massachusetts Institute of Technology (MIT). Kerberos is freely available from MIT, and made available in many commercial products (Basch, 1990/2008). Kerberos requires the use of secret-key cryptography much like that discussed under Pseudonymization. In Microsoft's security (Figure 7), the SSL technology is optional while Kerberos is a required security methodology, compared to Oracle (Figure 6) where SSL combined with key certificates is the primary security methodology.

Microsoft's instructions for publishing states that a review of connection settings, membership settings and other security setting on the Web server before publishing and to disable debugging, tracing, and custom errors after publishing (How To: Publish Web, 2005). Each vendor has their own means of securing their Web server(s). IBM suggests access scanning on demand using Nessus tool as shown in Figure 5.

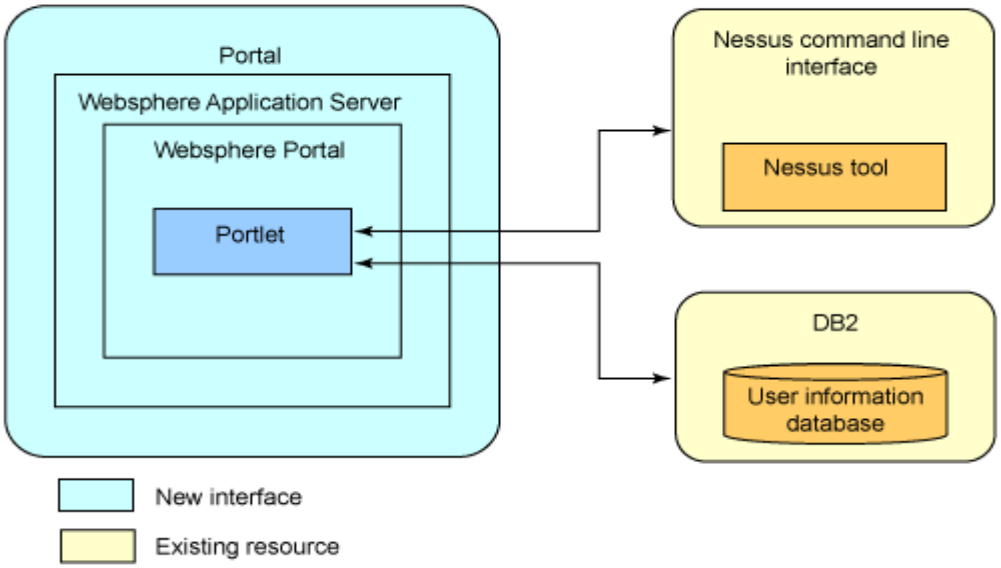

Figure 5 - Websphere On Demand Security Scan

(IBM Websphere Portal, 2006)

Oracle uses a blend of Secure Socket Layers (SSL), credentials and key encryption to secure their Web Server. The wallet as shown in Figure 6, manages public key security credentials. A public key is a value provided by a designated authority (Usually a database administrator (DBA) or systems administrator (SA).) as an encryption key that, combined with a private key allows a user, Web server access.

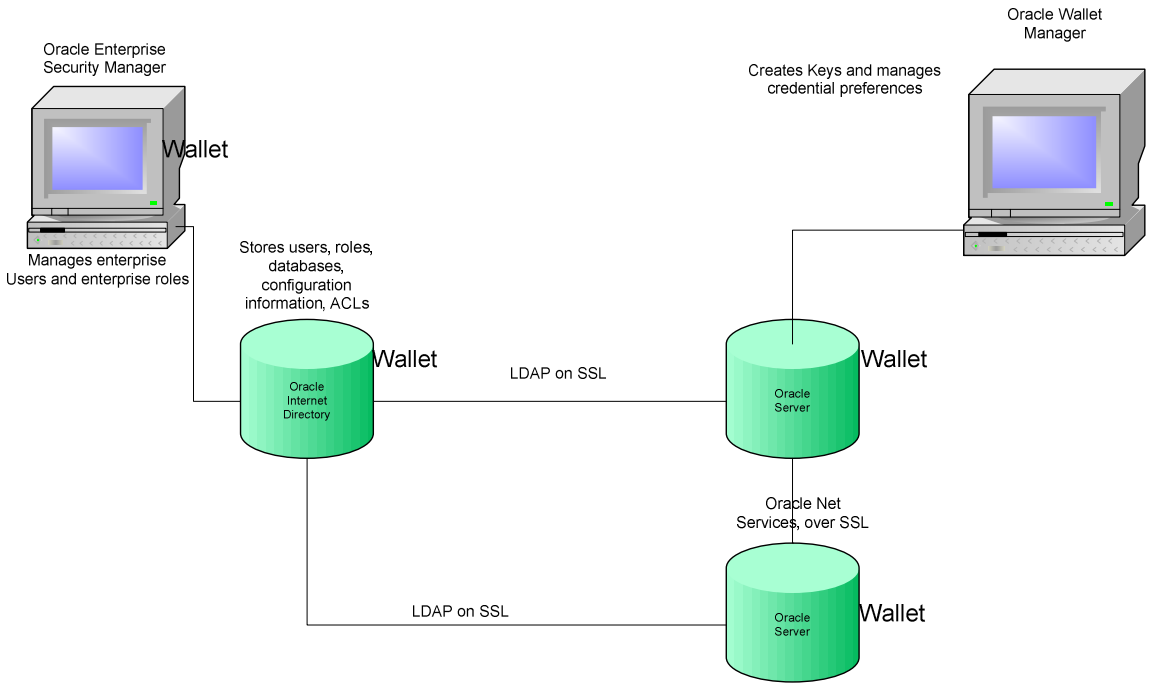

Figure 6 - Oracle Public Key Infrastructure

(Hale & Levinger, 2003)

Microsoft, shown in Figure 7, uses a combination of their own security products, combined with SSL or Kerberos or both.

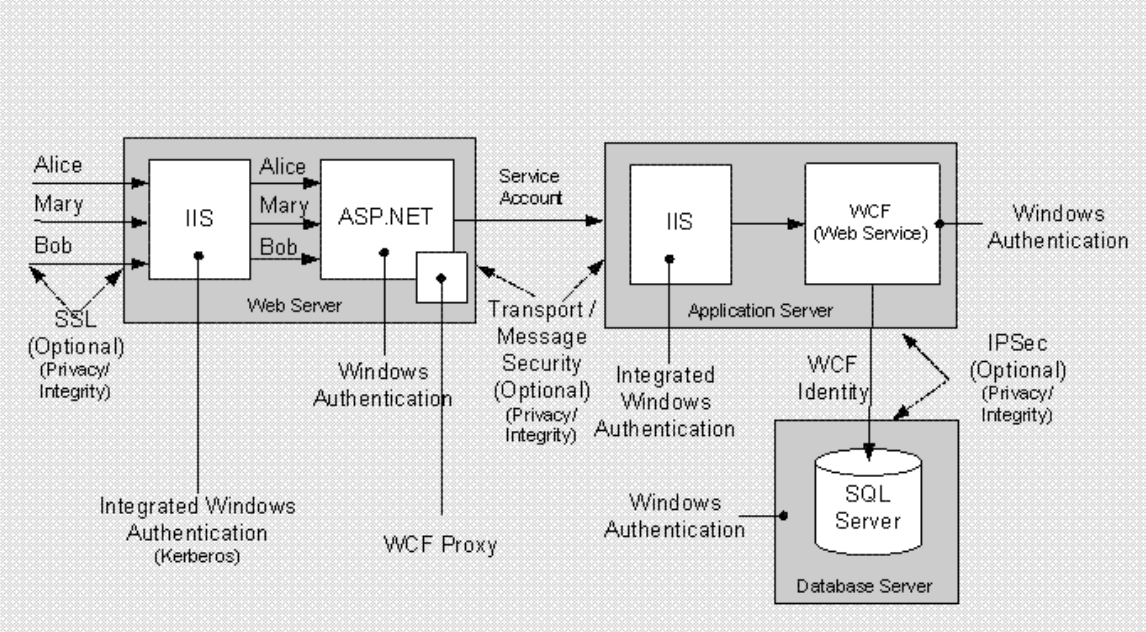

Figure 7 - Microsoft Web Security

(Meier et al., 2003)

Shepherd, Zitner and Watters recommend using different access rules for each user role (Shepherd, Zitner, & Watters, 2000) as do Oracle, SAP, Microsoft and IBM. Dependent on some technologies, access becomes available when users have matching keys or certificates. The question is what portal security is the best fit for the application and database designs, as it may come down to a combination of some of the above, or other recommended technologies used by specific vendors.

2.4.1 – Certificates and SSL

There are various schools of thought regarding using certificates to secure access between the end user thin-client and the Web server and the application. According to Jonathan Kames the advantages and disadvantages of SSL over Kerberos are:

1. Advantages
    a. SSL does not require an accessible trusted third party to establish a secure connection.
    b. SSL does not require the user to have a secret certificate or PIN.
2. Disadvantage
    a. If a system is compromised revocation certificates have to be circulated to all relevant servers and cached for a long time.
    b. Servers have to verify incoming user certificates against a revocation server and in that case, the revocation server must be highly available to a third party. In this case, the use of a third party eliminates the purpose of using SSL. (Kamens, 2000)

2.4.2 – Breaching of Network Information and Code Disclosure

While on any Web page, clicking on the browser menu option of *View*, and then on the menu option *Source* will allow the user to read the source code of that page in plain text. HyperText Markup Language (HTML), Extensible Markup Language (XML), and HyperText Transfer Protocol (HTTP) are the base of Web publishing and at the heart of thin-client technology. Compiled code in Java, C++, C#, Visual Basic or other application design products assists in stopping a malicious intruder from seeing private

information. The source may reference the application code by description such as, <script language="javascript" type="text/javascript"> but the compiled code itself is not readable. This is why Microsoft recommends that prior to publishing a Web application that a review of connection settings, and membership settings be preformed and to disable debugging, tracing, and custom errors after publishing (How To: Publish Web, 2005). The World Wide Consortium (W3C) is currently working on security documentation to cover the WEB industry best practices. Today they support using a combination of both SSL, Kerberos, and using AES as standards for WEB portal security (W3C, 2008).

Oracle, Microsoft, IBM, Novell, Apache, and other vendors, all have hardware and protocol solutions to help block intruders from obtaining information and code access however, such intrusions happen. The blocking of intrusion with hardware configuration, protocol settings and additional protection software goes beyond the limits of this study. Yet how to defend the application code and network information is an important issue to address, and should be part of every Web-based software planning.

2.5 – Summary of Security Practices

The standard for IS, is to first use user ID and password combinations to allow access to servers, applications, and databases. When using Microsoft's products the user ID and password is authenticated through the IIS and passed on to the .net application, which in turn passes the authentication through to the Microsoft SQL Server and database. If all medical organizations solely use Microsoft products this would simplify security procedures, yet this is not realistic. Many medical organizations have legacy database operations and other non-Microsoft servers, databases and applications in use.

Thus various security procedures are used to protect internal financial information and patient privacy information (Bunker, 2007), and user ID and password is only the beginning of security procedures.

The use of user roles and groups is the second link in the security chain as they help to define what each user can do while accessing the servers, applications and the databases. Application developers and DBA's design views, and perform encryption on data fields based on the user roles and groups (Christensen, Schneider, Thayer, & Winters, 2001/2003) and the SA's block user access to applications and databases based on user roles and groups and timed sessions. Since some legacy systems do not recognize or cannot recognize roles, user certificates, keys and data encryption become another link in the chain of security. However, SQL savvy personnel and people with malicious intent can breach these security processes. Thus, the audit logs and tables track possible breaches of security, and to help locate where the breach in security or the weakness in security take place (Swanson & Guttman, 1996/2008).

Chapter 3 – Security Methodologies

When E.F. Codd made his ground breaking speech in 1969 on the relational database model (Codd, 1969), and the use of SQL, he understood the importance of raw data and how to covert data to excellent information. Codd saw the relational database as data repository with strong table relationship and strong data normalization. Codd never discussed SQL as a programming language. Nor did Codd discuss how to use SQL to control data input, and to be procedural. Yet within only a decade, the relational database model became the foundation of database structure for vendors such as Oracle, Microsoft, and eventually IBM. SQL became more than a means for querying data as it also became a means for data manipulation. According to Casteel in her book *Oracle 10g SQL*, the problem with SQL is that it is open code and is easily read and interpreted. Anyone having access to the database can virtually see the database structure, read the code, change the code and manipulate the data (Casteel, 2006/2007). Casteel states, the best practice within the Oracle environment is to use Programming Language (PL), PL/SQL to create procedures, functions and triggers to hide the general SQL CRUD scripts from the general user. The term T-SQL or Transact SQL is the term used within other SQL environments and performs the same programming functions.

To go an additional step in order to protect the data, developers use complied code such as 4gl, Visual Basic, Java, C++, C# and a host of other software development tools to create front-end applications solutions to shield the SQL database completely from the end-users. As SQL has grown, developers have also incorporated additional means of securing data within the SQL database itself. No longer is the SQL database solely a relational model data repository.

When each user logs into an IS there is a cause and effect across a network. What the user views on the main menu is that which the IS personnel have programmed and developed for that specific user or user group. While some IS personnel subscribe to showing all menu options and disabling options for specific users and groups others create menu views specific to a user or group. According to Ricci, medical industry's best practice is to show only the menu items specific to a user, role, or group, thus limiting user knowledge of other network applications and databases. (D. Ricci, 2008) While this seems elementary, much work goes into developing the different views and testing user access to assure the user does not erroneously have access permissions to other application and databases (H. Singer, 2008). For end user ease of access to the applications and databases servers across the network, replicate the user ID and password. However, each server, application and database has a set of permissions for the user, role or group. Thus, recognition of each user, role, and group takes place and access allowed or denied by the servers. If the server has several applications or databases and the user or group has access rights to one of them, then the server allows access. With the use of user schema, it is up to application or database to recognize the user role or group and allow or deny access. Yet, there are probabilities of security breaches with limited knowledge of user ID, group and password.

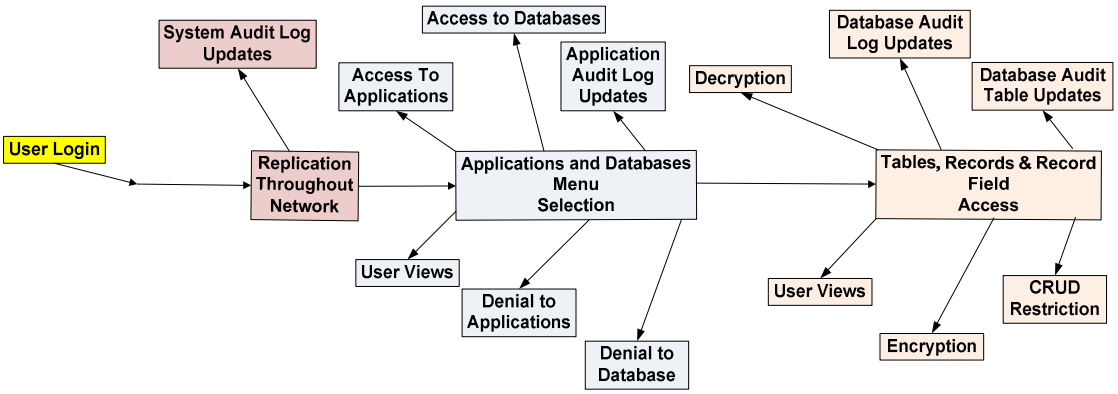

Figure 8 - User Login Cause and Effect

SQL databases contain schemas for user ID and passwords, roles and groups. Dependent on the organization users can log in directly into a database or through a front-end application. According to Casteel in her book *Oracle9i*, the problem with allowing users to log directly into a database is that they have better opportunity to manipulate data such as creating their own tables, copying data from one table to another, authorizing other user's access to their own created tables, and performing global updates (Casteel, 2003). User permission counters these types of activities and there are no rules, standards, or regulations stating that direct connectivity to a relational database through thin-client is unacceptable. However, it is highly recommended that a middleware application be the access point and a level of protection between the Web server and the database (Meier et al., 2003).

Within medical organizations, there is often a melding of technology. The billing database may be on an Oracle 10g PL/SQL solution, while the Siemens PACS is using Microsoft SQL 2008 and the patient history database is located on a legacy IBM Informix based mainframe. The systems administrator user ID and password does not have to be the same for each system, nor does the systems administrator User ID and password for one IS and environment have to allow full access rights the other IS's and environments

(B. Williams, 2008). With varying administrator passwords a database compromised by use of the administrator password and ID on one IS system, dose not necessarily compromise the data on the other IS systems.

3.1– Security Breach Beyond User ID or Group

Once a user ID and password authenticates allowing access to a network, applications, and database a probability of SQL injection and other malicious activity increases, developers must code and script security measures into their applications and databases. According to Bunker in his article, *Is Your Privacy Protected?* malicious activity is a process of invention and it is very difficult to anticipate every possible security breach (Bunker, 2008). This is why audit logs and tables become very important for reconstruction of events. Using the information obtained from the logs, developers of SQL databases and applications can build in additional security measures (Swanson & Guttman, 1996/2008). However, planning and taking precautions during the design and development phases to counter the known malicious activity is the best strategy (Bunker, 2008).

3.1.1 – Limiting the Possibilities for SQL Injection and Malicious Activity

SQL injection attacks occur when SQL commands inserted into a field of entry, such as the User Id field, reveals or manipulates data other than the fields intended use. This can be one SQL command or procedural SQL scripts, dependent on how many characters the field allows. As an example, one of the mean for gaining access to a database is for a hacker to enter into the user ID field the SQL statement below.

‘UNION SELECT id, name, ‘’, 0 FROM sysobjects WHERE xtype = ‘U’

Due to the flexibility of SQL, the SQL interpretation to the above line is as a selection request for all users from the user table.

'UNION SELECT 0, UserName, Password, 0 FROM users –

The SQL response is a list of the data contained in the users table of user ID's and Passwords. The hacker can then select any one of the user ID and password and then gain access to the database. According to Jansen in an article written for the NIST, *Computer Security*, in this situation, the developers will want to restrict the use of specialized characters and length of the field (Jansen W., T., & Scarfone, 2008). What presents a problem is that in limiting specialized characters such as the apostrophe (') or hyphen (-) affects the ability to use the proper form of user names, such as jo'day01 or john_o'day.

If a user ID and password is a known element then the hacker only has to insert SQL into fields of entry screen to change or manipulate data. According to Litwin, three features make SQL injection attacks quite powerful:

1. With the use of hyphens the hacker has he ability to embed comments within a SQL statement and into a data field.
2. The hacker has the ability to alter or destroy data by stringing multiple SQL statements together, and to execute them in a batch.
3. Using SQL to query metadata from a standard set of system tables to obtain or alter data (Litwin, 2008).

Litwin states, the best defense is limiting the number of characters allowed in data fields (Litwin, 2008). By limiting the number of characters, SQL injection is difficult and near impossible because the SQL statement truncates to the character length restriction. Thus

the statement 'UNION SELECT id, name, '', 0 FROM sysobjects WHERE xtype = 'U' on an eight character restriction truncates to 'UNION S which the application will either deny user access or throw an error.

According to Jansen, another concern of SQL injection is that some organization will use multiple interpreters in tandem with other applications. A harmful operation can unobtrusively be passed through the IS from one database or application to another compounding the problem.

3.1.2 – Limiting User by Views

Using the data fields from Table 1, Chapter 1, Figure 9, reflects an example for planning and development of the user views. As a hypothesis, assume that an individual with malicious intent has found a means to log into a medical application or database. Depending on the identification assumed is what information presents itself in either a view, or a SQL inquiry. According to Singer, while views and field encryption keep the honest employee, honest, the truth is, that anyone with malicious intent will still be able to obtain private data dependent on the user ID and group permissions used. If the intruder breached the administrator user ID and password, then all data is available within that database (H. Singer, 2008) In his article, *Oracle Database Security*, Nathan Aaron addresses theses issues. Aaron states poor database view designs, poor user access schema design, and placing too much trust in employees regarding data access can be the cause of users becoming dishonest (Aaron, 2006).

As previously discussed, a user with the appropriate privileges can create a table and give another user access privileges to the table. Thus using the planning views and encryption in Figure 9, should the admissions user have the privilege to create a table and

gives a caregiver rights to the newly created table, the caregiver now sees all of the fields that are denied under the care giver group.

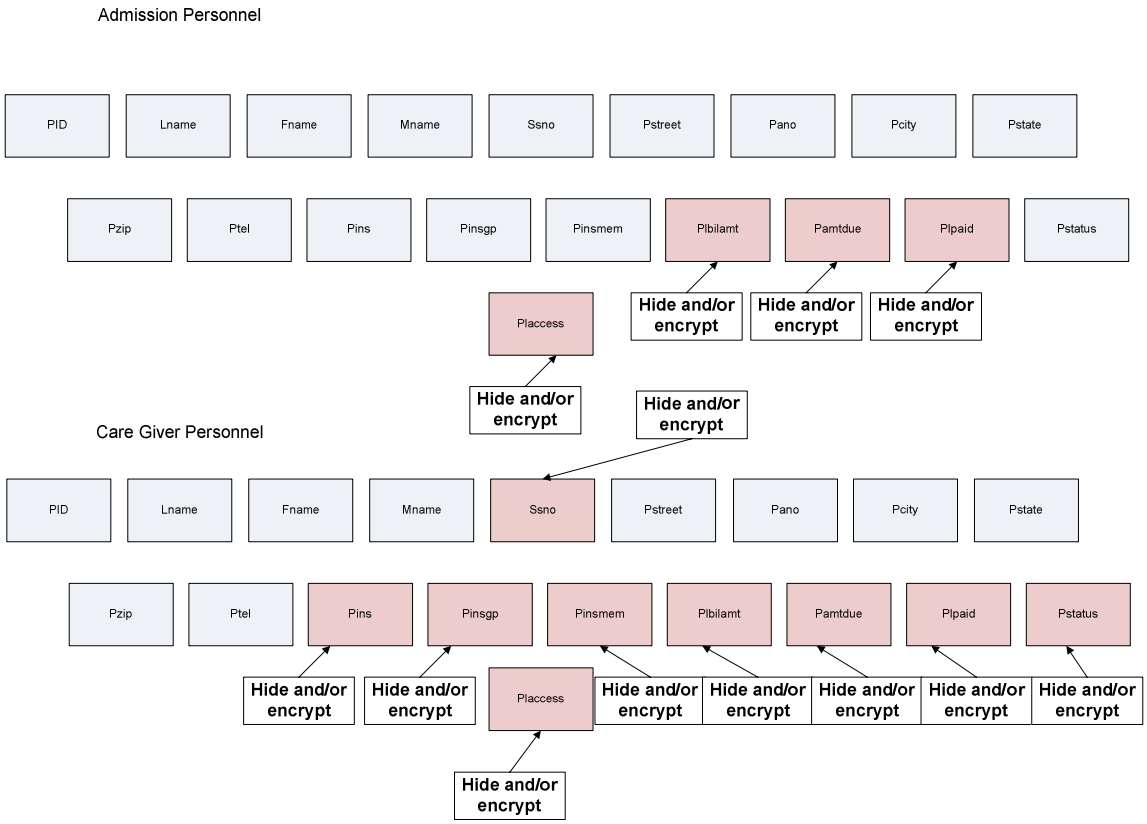

Figure 9 - Planning Views and Encryption

Greenwald in his book *Oracle Essentials,* also states that planning and testing of user views in combination with user access rights is an a important practice of DBA's (Greenwald et al., 1999/2004). Both Aaron and Greenwald note that allowing users the rights to create tables and assign other users privilege to their tables are security wise problematic.

3.1.3 – Employee Security Breach

According to NIST, security breaches do not always come from the outside. Whether it is a disgruntled employee, an employee in financial difficulty attempting to make extra monies selling information, or just by shear accident, an employee can often find a way to access data and information that they are not privy to, or information needed to perform their job function. This is where role specific views, or database or data field encryption are useful. However, the update of user ID with a date and time stamp into a record when any CRUD activity occurs is an additional security measure (Swanson & Guttman, 1996/2008).

As a new supposition, assume that several patients have complained that they have recently been contacted by an insurance carrier representative, who has seemed to know their information regarding current insurance carrier, the dates of their admissions for medical care, and worst, has ran credit checks based on their social security number. SQL queries ran on the billing database, patient table, on the *last accessed* field of each record, can quickly show the count of records accessed on various dates or date ranges by each end user. According to DeLuca, A spike by one user compared to other users could be representative of a user misusing their position and their access to secure patient information. Some medical organization run scheduled SQL queries weekly or monthly, on this type of activity to note end user access anomalies (T. DeLuca, 2008). Comparing the user's activity to audit logs and tables will provide additional information such as, were the records accessed for read only purposes or were updates made and type of updates.

3.1.4 – Troubleshooting with audit logs and audit tables

Rebecca Sausner, in an article from the Bank Technology News *Keeping Your DBA Honest* notes that when George Clooney was hospitalized, that a review of the hospitals database audit logs noted every doctor and nurse who violated patient trust by celebrity snooping on his medical records. However, if the DBA was the one snooping, he knew how to cover his tracks (Sausner, 2008). While third party software is not a part of this study, it is important to note that third party electronic signature software exists, that tracks audit logs and notes if a breach of a log occurs.

The use of application and database audit logs show errors within the application, process timing, time stamps of events, and user authentication changing from the user to a system or administrative user. The audit logs are invaluable for troubleshooting and for noting security breaches (Swanson & Guttman, 2008). The user SQL logs reflect the actual SQL scripts and the application is reflecting the application scripts.

A sample user SQL audit log may look like the following:

jsmith01 10:35:16:08:15:2008 SELECT * FROM billing WHERE pid = 12345

jsmith01 10:35:16:08:15:2008 SUCCESS 1 row returned

The corresponding application audit log may look like the following:

jsmith01 10:35:16:08:15:2008 received from frm_patientR patient_id = 12345

jsmith01 10:35:16:08:15:2008 service call get_patient

jsmith01 10:35:16:08:15:2008 patient_id 12345 found

jsmith01 10:35:16:08:15:2008 frm_admsView refresh ss_no encrypt

Where the application is concerned several security measures are in place, the paient_id is an alias for the database field pid. A procedure get_patient contains the

compiled code to access the database, and possible a call to a database procedure thus hiding the database name, and the SQL table name within the database. Record retrieval to the application shows the end users assigned view that encrypts the social security number. No one reading the application audit log will gain knowledge of the database, unlike user SQL audit logs that record ever SQL script ran and provide information regarding the table accessed and the field name.

Unlike the SQL server in which audit logs must be located on the server, the user SQL and application audit logs, can write to alternate locations (Howie, 2002). According to Williams, in the thin-client environment, the last place that one would want the user SQL or application audit logs to write is to a workstation hard drive. The SQL logs and the application logs grow very quickly in size, if left on their respective servers they could easily interfere with processing and space allocation. The industry best practice is to select an obscure server for writing user SQL and application audit logs. This has a drawback in possible loss of processing speed and in maintenance (B. Williams, 2008).

3.2 – Database Design Security

As previous noted many security methodologies overlap between the database and the application. However, whether planning a SQL database from scratch or implementing an existing SQL database into the thin-client environment there are SQL database security methodologies to consider beyond the user ID and password, groups, logs and views.

According to the NIST, use of audit tables to note changes to a data record by inserting the original data, the changed data and the user ID with date and time stamp is a

security methodology to consider (Swanson & Guttman, 1996/2008). The NIST also recommends creating audit tables that record Dynamic Link Library (DLL) changes in activity such as table creations, table reconfiguration, and table drops. When this type of DLL call takes place, an insert into an audit table takes place with the table information and the user ID with date and time stamp (Singhal, 2007). Additional database security options are:

1. Use of AES data encryption standardization for encryption of a SQL database
2. Use of AES data encryption standardization for encryption of a data record field
3. Use of procedures, functions and triggers called in scripts by name
4. Limiting the data field input through the use of forced data selection
5. Limiting by number data field entry characters
6. Disallowing special character entry within data fields
7. Restricting user session times to work hours
8. Restricting users from table creation or table drops
9. Restricting users from table sharing (Kiely, 2005)

3.2.1 – AES Data Encryption

As previously discussed, AES data encryption is the current standard of NIST (Smid, 2001). However, there are performance issues with the use of encryption (Riedl, 2008). Image databases such as PACS would not only use a lot of equipment processing time to encrypt and decrypt but also, be a hindrance to a user waiting for the decryption process to complete (Riedl, 2008). Yet, the use of encryption and decryption on character data is expedient (Smid, 2001). Encryption of the database or only selected data fields of each table are encrypted depends on the sensitivity of the data they contain and how the

data is linked to the application and the Web portal. When a SQL database directly links to a Web portal, IT industry best practice is to encrypt the database (Shepherd et al., 2000). If a middleware application shields the database then encryption of sensitive data fields is an acceptable practice (Smid, 2001). Configuring a database, data table, or data field for encryption, results in the objects encryption until a user retrieves the data. Decryption of data occurs when the user's access privileges give the user access to the data to perform CRUD activities. If the user has access privileges to some of the data but not all, the items that the user does not have access to will remain encrypted.

3.2.2 – Procedures, Functions and Triggers

Casteel in her book, *Oracle 9i Developer* explains the expediency of programming PL/SQL procedures, functions and triggers due to compacting code in one specific script that called by name by another SQL script, performs SQL calculations and CRUD activities. Considered an IS industry best practice, the writing of procedures, functions and triggers enhances the functionality through reuse of scripts and not displaying SQL activity to the end user (Casteel, 2006/2007). Powell in his book *Oracle 10g Database Administrator Implementation & Administration* takes this a step further explaining that the use of procedures, functions and triggers are security features to protect data integrity and the functionality of the database from the users (Powell & McCullough, 2007). The use of procedures, functions and triggers within a database allows for the developers of a middleware software application to call each by name, thus adding another layer of security between the application and the database by not revealing table name, field names, or database functionality information from within the application (McGovern et al., 2004).

3.2.3 – Limiting Data and Disallowing Special Characters

While discussed earlier, the methodology of limiting data entry and disallowing special characters is used to thwart malicious activity such as SQL injection, however there are other security reasons to limit data, and stop users from using special characters. Controlling fields of data by drop down boxes, radio box selection and field length maintains data integrity. While data selection will not stop the occasional selection of *Mr.* when *Mrs.* was intended, it will stop other typographical errors that make it difficult to perform accurate data extrapolation to obtain information. Data restriction makes it difficult for anyone to change data globally within the database, thus the inability to change all records to *Mr*. (Powell & McCullough, 2007).

Special characters such as the asterisk (*) have meaning within the SQL language. The asterisk in SQL means *everything*, thus having a field containing an * can make it difficult to extrapolate data by interfering with the original SQL Select statement (Casteel, 2006/2007). As discussed under SQL injection limiting special characters and restring field length stops a user from entering SQL commands such as, *Update patient Set ptitle = Mr.,* into an entry field and globally changing every patient file, *ptitle* field, in the patient table to *Mr*. (Jansen W., T., & Scarfone, 2008).

3.2.4 – Restricting Users

A concern of DBA's is restricting users to such a point that they are rendered incapable of performing their job function, and is why testing user access is extremely important (Powell & McCullough, 2007). However, it is also extremely important to assure that user access is not manipulated or misused. Restricting user session access to their working hours, sharing user created tables, and the right to create tables is a high

security consideration (IEEE, 2006). Restricting users falls under the user schema as discussed earlier and noted here solely as an accompaniment to database security.

3.3 - Application Design Security

Meir, in an article on design patterns on Microsoft's Web site states five security practices to make when designing a Web middleware application:

1. Assume all input is malicious.
2. Centralize your approach.
3. Do not rely on client-side validation.
4. Be careful with canonicalization issues.
5. Constrain, reject, and sanitize your input (Meier et al., 2003, p. 4)

An example of malicious input is SQL injection as discussed above, and thwarted by limiting the number of characters allowed into a field.

3.3.1 – Malicious Activity

Some data fields cannot be restricted to a small number of characters. What if the field of entry is a narrative field such as a patient diagnoses? When writing the code for the medical Web application, directing an insert of HTML text tags before and after any narrative data field will render the malicious code to text and strip it of functionality. Anyone with malicious intent inserting a SQL command such as, *DELETE * FROM patient* into a data field, results in an insert into the database *<text> DELETE * FROM patient< /text>,* and not the deletion all the data contained in the SQL database table named patient (Meier et al., 2003). As discussed under SQL Injection rejecting special characters such as: %, --, //, < >, =, +, ∑ is another option, however within the medical

environment this may be too restrictive for the clinical end users who need to comment or reference scientific formulas or data, thus the use of HTML tags is also the recommendation of the W3C (W3C. 2008).

3.3.2 – Canonical Restraint

As previously discussed, layered n-tier OOP application design allows for writing code by reuse of patterns, thus making the process expedient and consistent. According to Fowler in his book *Patterns of Enterprise Application Architecture*, poor programming practices are hard coding of path locations, specific data values, and pointers to specific server names (Fowler et al., 1963/2003). Yet, what if part of a user's responsibility is importing data from external data source, such as the cost of medical supplies from a supplier, or insurance payment rates for procedures? For our discussion purposes, we will state that the import file downloads to a secure server and a specific directory. Rather than hard coding the path into the application, create a SQL system data table, designed to contain paths and pointers. The form view references the SQL systems table obtains the correct data location, limiting the user to a location, yet allowing the user to select the appropriate file to import. This process works in conjunction with user access and logins, and the user must have permissions to the secure server, directory and file (Meier et al., 2003).

3.3.3 – Restricting Users

Restricting user menu selections, limiting the information seen by creating user role specific views, and designing field drop down boxes for restricted field data selection are tools that aid in keeping data accurate and secure. According to the IEEE, within the

thin-client environment, these tools become more important to data security (IEEE, 2006), as they limit the possibilities for misuse and incorrect data entry.

Calling database procedures rather then using SQL CRUD scripts within the application code, hides the identity of the database and the SQL processes from the end users adding an additional layer of security. According to Watson, limiting the calls for sys or sysadm roles and making sure the user's credentials authenticate throughout all application and database processes maintains the user roles security. If possible, limit the session times that a user can access the application to their regular working hours (Watson et al., 2005).

3.2 – Summary of Methodologies

While there is overlapping of security methodologies between the SQL database and application Table 2 below depicts the methodologies discussed. An important note is that some methodologies work in tandem or individually, either within the database or the application dependent on the technologies used and the data location. As an example, database encryption would eliminate the use of field encryption within the application. Yet, if the application pulls from two data sources such as a SQL database and a legacy database, field encryption may take place at the SQL database while the legacy field encryption takes place at the presentation layer of the application.

Table 2 - Security Methodologies

Security Methodology	SQL Database	Application

Use of User ID and password validation for IS authorization.	Yes	Yes
Creating user views based on user roles.	Yes	Yes
Configuring audit logs on server.	Yes	Yes
Creating database and application audit logs	Yes	Yes
Limiting character entries to an appropriate length, for the data contain in the data fields.	Yes	Yes
Eliminating use of special characters that SQL would interpret as a command function	Yes	Yes
Designing data entry to insert HTML tags to render malicious SQL code injection harmless	No	Yes
Limiting field entry to drop down selection items that are specific to the type of data requested and not left to user input.	Yes	Yes
Limiting user menus selections to only those applications and databases the user needs in order to perform their job function.	Yes	Yes
Saving user database and application log activity to an obscure server	Yes	Yes
Creating audit tables within the database to track CRUD activity of private information	Yes	No
Creating audit tables within the database to track table	Yes	No

DLL activity especially the activity that pertain to table creation and table drops.		
Use of procedures, functions and triggers called by name rather than general CRUD statements	Yes	Yes
Use of field name aliases to conceal database tables and fields	No	Yes
Use of AES encryption to conceal data contained within a database or data fields.	Yes	Yes
Limit user sessions to business hours or work schedule to prevent illegal access during non-business hours.	Yes	Yes
Controlling access by authorized users by using SSL, Kerberos, or key certificates	Yes	Yes

Using all 18 methodologies within a thin-client environment between the SQL database, its software application and Web Server interface is very probable. Dependent on the technologies in place, the use of data extrapolated from legacy systems, and new SQL database design vs. older SQL database design determines where and how each is applied.

Chapter 4 – Analysis and Results

While stated in Chapter 1 that the medical industry did not have a history to assist when planning security to meet the HIPAA and SOX regulations, today it does (D. Ricci, 2008). IS security best practices developed over the past ten years through lessons learned, and with the aid of IEEE, NIST, W3C, and OITS have assisted with the development of the 18 methodologies. However, how to deploy the security solutions and in what circumstances is left to IS personnel to plan, design and implement.

4.1 SQL Database Security

As previously discussed, due to the risks of exposure through Web technologies there are issues to consider when planning SQL database security within the thin-client environment. Table 3 below depicts the planning questions and comments

Table 3 - Database Security Considerations

Question	Response	Comments
Is this an existing SQL database?	Y/N	Name of database
Is this a new SQL database?	Y/N	Name of database
Use of a middleware application	Y/N	Name of application
AES database encryption	Y/N	
AES table record field	Y/N	Names of tables, and fields for encryption

encryption		
User Views	Y/N	Naming convention of user views
Procedures	Y/N	Naming convention procedures
Functions	Y/N	Naming convention of functions
Triggers	Y/N	Naming convention of triggers
Audit SQL logs	Y/N	Naming convention of logs and storage path
Audit Tables	Y/N	Naming convention of tables, and the field names of data to record
User rights to create tables	Y/N	User groups with rights
User rights to share tables	Y/N	User groups with rights
Session restriction	Y/N	User groups with restrictions
Smart card access	Y/N	
Key Fob access	Y/N	
Certificate access	Y/N	

4.1.1 – Planning Security for Thin-client Implementation with an Existing SQL Database

Planning security for an existing SQL database involved many of the same processes that are involved with the creation of a new SQL database. An additional item of consideration is the ramifications of possibly having to alter the security features within the database. Other tasks and considerations are as follows:

1. A review of each table, the data it contains and planning for the CRUD activities of the end users as in the billing table reflected in Table 1.

2. Data mapping user roles and groups as in Figure 9, to note what fields require data encryption (if any) and the screen views for users.
3. Review of the data contained in the SQL database and making the decision to encrypt the entire database due to the sensitive nature of the database.
4. Deciding if a middleware application would better secure the database and if so:
    a. Are the views and encryption going to be within the application or the database?
        i. If so, consideration on how to remove the current security features of views and encryption from the current database without affecting the users and the reliability of the database.
        ii. If not, consideration on how to interface the current security features with the application without affecting the users and the reliability of the database.
    b. Are the current triggers, functions and procedures naming conventions within the database consistent or do they required revisions?
    c. Would the interface to application require additional triggers, functions, and procedures to secure the database?
5. Are there fields for data entry that can be restricted to drop down selections?
6. Are there fields for data entry that can be restricted in length?
7. Are there SQL audit logs, and if so, where are they stored?
8. Are there the current audit tables within the SQL database sufficient?

Although the SQL database is in production and functioning, the review will help to assure HIPAA and SOX compliance. While the database may have met the

requirements under fat-client technology, the exposure to the Web and using thin-client may require additional security features. As an example, currently some user permissions allow session table creation to obtain information for clinical trials. While the information the users obtain is not patient specific or may use a patient identification number with no name associated, the ability to create the table and access data tables directly using SQL commands presents a security issue within thin-client (Greenwald et al., 1999/2004). Possibly a new reporting feature, or a new view to extrapolate patient related, yet not patient specific information is created, using the users criteria, rather than allowing session table creation. The design would call procedures and functions by name rather than directly perform SQL commands, thus adding another layer of security. The premise is not to make the end-users job more difficult; it is to comply with SOX and HIPAA regulations to secure data. As long as the reporting or view offers the end user the information sought in the performance of job function then the feature meets both security and user's needs .

4.1.2 - Planning Security for Thin-client Implementation with a New SQL Database

As a fellow student in my Database Concepts class, David Kizhner, so aptly stated during a discussion on data normalization "It is about the data, the whole data and nothing but the data, so help me Codd" (D. Kizhner, 2007). Security is all about the protection of data, SOX and HIPAA are all about the protection of individuals' sensitive personal data. Considered to be an IS industry best practice is to include security features within each phase of design, development and implementation (McGovern et al., 2004). Knowing that the database implementation will be within thin-client technologies, makes planning for security more focused and does not change the questions asked as outline in

section 4.1.1 except in the form of *how* are these features are implemented, not *are* these features currently implemented.

Knowing that the database will have a front-end application does beg the question as to whether the database will control the AES encryption, or if application will use AES encryption. The same applies to *Views*, *Triggers Procedures* and *Functions*. Again, there are no hard rules stating that one is a better practice over the other (Monroe, Kompanek, Melton, & Garland, 2003). Using one security feature within the database and another within the application does not mean that one or the other is a best practice. However, consistence in design is an IS industry best practice and is extremely important for future changes over the life cycle of the database and its middleware application (Powell & McCullough, 2007). The ramifications of incorporating one security feature within the database and another within the application are considerations. While in certain circumstances, it may be best to use AES encryption at the database level, or fields within the database, it may be best to design views and limit field character length at the application level.

4.2 – Application Security Design

As within the database design the question involving the application design revolve around the whether or not the application is an existing application with an additional feature added or if this is a new application design for the use of thin-client technologies. The processes of questions are slightly different due to the knowledge that this application will access the SQL database and work with the database to manipulate data. Thus, many of the database features are a consideration.

Table 4 - Application Security Considerations

Question	Response	Comments
Is this an existing application?	Y/N	Name of application
Is this a new application?	Y/N	Name of application
Will the application use aliases to access the database table and fields	Y/N	Name of database Naming conventions of aliases
Will the application use MVC to access the database?	Y/N	
Is AES record field encryption done at the database level?	Y/N	Names of tables, and fields encrypted – user, roles and groups affected.
AES record field encryption	Y/N	Names of tables, and fields for encryption – user roles and groups affected.
Are database views used?	Y/N	Names of views and purpose of views – user, roles, and groups affected
Are application views used?	Y/N	Naming convention of views – user, roles, and

		groups affected.
Are application procedures used?	Y/N	Naming convention of procedures and purpose of procedures
Are database procedures used?	Y/N	Procedure names, and when to call the procedure
Are functions used?	Y/N	Naming convention of functions and purpose of function
Are database functions used?	Y/N	Functions names, and when to call the functions
Triggers	Y/N	Naming convention of triggers and when to call the trigger
Are the database triggers used?	Y/N	Trigger names, and when to call the triggers
Field length restrictions	Y/N	Names of fields and restriction
Field entry type restrictions		Names of fields and type restriction
Application logs	Y/N	Naming convention of logs and storage path
User access by user, role and groups	Y/N	Naming conventions
Session restriction	Y/N	User groups with restrictions
Smart card access	Y/N	
Key Fob access	Y/N	
Certificate access	Y/N	

While stated in section 2.3.1 that hiding the database behind database handler routines, is old technology but allows the database design to change while preserving the original interface (Britton & Bye, 2004), it is important to note that this is considered one of the better security practices. Using the applications to call by name the database procedures and functions minimizes the risk of code disclosure (Swanson & Guttman, 1996/2008).

4.2.1 - Planning Security for Thin-client Implementation with a Middleware Application

One of the medical industries problems with implementation of new technologies is the existence of many legacy IS systems (Bunker, 2008). When faced with adding new technologies, issues and problems occur due to forced restrictions or constraints because of the legacy systems limitations. Questions occur regarding programming consistency, such as; since one of the middleware application modules must use AES encryption due to the legacy systems constraints, do all application modules thereafter have to use AES encryption at the application level? There is no rule that states that developers must continue to use old technology within new projects due to the limitation of legacy systems (McGovern et al., 2004). The important fact is to be consistent with the programming practices within the same programming module. Thus, if using AES encryption at the database field level, then throughout the life cycle of the database and its middleware application module, all AES encryption takes place at the database. IS industry best practice is to note security features within the database and to build upon security within the application and not duplicate (McGovern et al., 2004). Using the questions in Table 4, if the answer to AES field encryption at the database level is yes,

then there is no reason to use AES encryption within the application. The same may or may not hold true for views.

The views at the database level may be strictly for users directly accessing the database outside of the thin-client technology. The views constructed within the database may not meet the criteria for restricting views within the thin-client application. If the criteria for the thin-client environment specifically state that no user has access directly to the database, then the requirement calls for application designed views. Thus, it is probably to have views in both the database and within the application (Meier et al., 2003). What is important is that the views meet the security requirements of the user, role, and groups. Another scenario is that an application view calls a database view that is *read only,* thus meeting the user's needs to view certain data without directly manipulating the data within the database, and meeting the application security of using a named object call to the database which shields the original source from detection.

The use of the database functions, procedures and triggers greatly reduces application programming time and the chance of misinterpretation of the database functionality (Powell & McCullough, 2007). While the application may have some procedures for database access and other application functionality, using the database functions, procedures and triggers to manipulate the data aids in maintain data integrity and security (Powell & McCullough, 2007). The concern is using the correct function or procedure within the development of the application. It is extremely important for database developers and the application developers to have constant communication regarding the design, development and implementation of thin-client technologies to

avoid duplication of effort, maintain consistency in design, and the reuse of code for expediency (McGovern et al., 2004).

Dependent on technology in place there are times where a duplication of security features occur such as:

1. Using key certificates to access the application and the database
2. Using Key Fobs and Smart cards to access the application and the database
3. User ID and Password verification to access the application and the database (Riedl, 2008)

While this duplication may go unnoticed by the end-user, validation passes between the technology, the application and the database. The application and the database should each have a user access schema with user role and group permissions. The server and the application audit logs note user access times.

The application audit logs should contain more than just the user's access. The purpose of the audit logs is to trail every action that a user takes while working within the application. The audit logs can be an excellent troubleshooting tool for application errors however, more significantly, they serve as a record for any attempted security breach as well as a record of a security breach. By reviewing the audit logs and noting how security failed, developers can then correct the hole within the application security (Swanson & Guttman, 1996/2008).

4.3 – Summary of Security Implementation

Securing data within the thin-client environment is a combined effort of the database and application developers and the systems technology personnel (McGovern et al., 2004). While there may be overlapping security features such as user ID and

password for access at server, the application and the database there are security features that have no need for overlapping or require duplication. The most important part of security design is consistency of design, for doing so eliminates the possibilities of security holes, security misinterpretation, and user access conflicts (Christensen, Schneider, Thayer, & Winters, 2001/2003). Thus if encryption is used at the medical SQL database level or database table field level, then there is no need for encryption at the application level.

While there may be reasons to have user, role, and group restrictive views within both the database and the application, proper design of database views allows the application developers to call the views by name and reduce the risk of code disclosure. This premise also holds true for database procedures, functions, and triggers (Meier et al., 2003). On the other hand, application developer's use of alias table name and table field insure that the code does not disclose any pertinent information on the database. This is important within the thin-client environment due to exposure through the Web interface and the ingenuity of those with malicious intent (Watson et al., 2005). Anyone gaining intimate knowledge through malicious activity of the application, the database or both presents an immediate concern for data security and integrity (Britton & Bye, 2004).

Restricting users by limiting field character length, denying special characters, and forcing data selection for data field entry guard against SQL injection and malicious code entry and protect data integrity. While these restrictions are at both the database and application level it is beneficial to be consistent with designing techniques to eliminate confusion over the life cycle of the database and the application during future changes (McGovern et al., 2004). Restricting user access session times to work hours aides in

eliminating intrusions during off hours and through audit logs alert DBA's, and IT personnel of an attempted breach using a users login ID and password (Watson et al., 2005).

The creation of SQL audit logs, SQL audit tables, and application audit logs are important security features and an IS industry and medical industry best practice. The audit logs and tables are a valuable source for noting security breaches and attempted security breaches and assist developers in shoring up both the database and application security against the malicious activity (Swanson & Guttman, 1996/2008).

Chapter 5 – Lessons Learned and Next Evolution of the Project

The similarities in designing security for a medical SQL database and a middleware application became very apparent quickly within the researched documentation. The question then became, when would using one security design within the medical SQL database or middleware application be better within the overall design? My conclusion is that it more important to be consistent within the security design and have a documented security design. Thus, if using AES at the medical SQL database level would then require the use of AES with all medical SQL databases where data encryption is a requirement. The reasons that consistency is so important, is over the life cycle of the SQL database and its middleware application is that:

1. Consistency in security design makes changes, updates and revision easier by knowing the security pattern and thus aides in maintaining security integrity.
2. Consistency in design makes it easier for unit testing by following the security chain through the SQL database, its middleware application and server interface.
3. Consistency in design makes it easier to troubleshoot, and to shore up a security design in the event of a breach.

Communication

Because the SOX and HIPAA regulations and the need to maintain data integrity and confidentiality forces the designers, developers and engineers to review every data process taking place within the IS, and to document and evaluate security operations on a regular basis. This also means that the DBA's, designers, developers and system engineers have excellent and open communication among themselves. The security

design is too important not to have an understanding between all parties responsible for the security of confidential information. Without proper communication between security development teams, a change within the database, the application, or at the server level can break the security chain throughout the IS.

System Documentation and Audit Logs

Audit logs and audit tables are extremely important to the successful security design. In the event of a breach of security, following the audit logs and tables enables the designers, developers and engineers to focus directly on the area where security failed to meet the SOX and HIPAA requirements and the organizations security requirements.

While not within the realm of this study, that use of external third party security programs and utilities are items for consideration, and evaluation for use. Often developers attempt to reinvent rather then to use a documented and useful solution is already in existence (Fowler, 2003). This is an area for further investigation and interpretation in security design.

Testing and Evaluation

As with database design and application design, an important aspect of security design is the on going evaluation and testing of the security features. Security design also has a life cycle that includes updates and revisions as needs change through database and application updates and revisions. While not a part of this study, it would be interesting to investigate testing methodologies.

Summary

The five lessons learned are the need for consistency of design, open communication with all responsible security personnel, excellent on-going

documentation through logs and audit tables, an understand the life cycle of the security design and the regular testing of security features. While there is cross over between database and application security design it is important for every organization to review what is currently in place, what is working well or what should be improved upon, and to determine what features are best utilized within the database opposed to use within the application.

Within the thin-client environment, it is extremely important to know the table and field structure of the SQL medical database, how each piece of data relates to the SOX and HIPAA rules and regulations, and to the business rules of the organization for planning security. Knowing the database structure and the value of the data as it relates to regulatory compliancy, makes it easier to evaluate user roles and groups for the security design of database and application access, user views, database and application procedures, functions and triggers. Due to the exposure to the World Wide Web (WWW) and the probability of malicious activities, that the methodologies discussed may not be enough to protect data and the use third-party security utilities are a consideration.

References

A guide to the Sarbanes-Oxley act. (2002). Retrieved April 28, 2008, http://www.soxlaw.com/index.htm

Aaron, N. (2006). *Oracle database security*. Retrieved August 18, 2008, from Infosecwriters Web site: http://209.85.173.104/search?q=cache:B-QWgf08WKIJ:www.infosecwriters.com/text_resources/pdf/Oracle_NAaron.pdf+Database+View+Security&hl=en&ct=clnk&cd=27&gl=us

Andrews, C. (2001). *SQL server 2000 security secrets*. Retrieved July 11, 2008, from Microsoft Certified Professional Magazine Web site: http://mcpmag.com/features/article.aspeditorialsid=210

Barak, B., Herzberg, A., Naor, D., & Shai, E. (2002). *The proactive security toolkit and applications*. Retrieved June 15, 2008, from ScientificCommons Web site: http://de.scientificcommons.org/492942

Basch, R. (2008). *What is Kerberos*? (Original work published 1990) Retrieved August 2, 2008, from Massachusetts Institute of Technology Web site: http://web.mit.edu/Kerberos/#what_is

Britton, C., & Bye, P. (2004). *IT architectures and middleware* (C. Britton & P. Bye, Eds.). Boston: Pearson Education, Inc.

Bunker, G. (2007, September 10). Is your privacy protected? *Hospital Management*. Retrieved May 28, 2008, http://www.hospitalmanagement.net/features/feature1311/

Christensen, M., Schneider, G., Thayer, R., & Winters, J. (2003). *Software engineering* (R. Thayer & M. Dorfman, Eds.). Danvers, MA: Wiley-Interscience. (Original work published 2001)

Christensen, M., & Thayer, R. (2001). *The project manager's guide to software engineering's best practices*. Piscataway, NJ: IEEE Computer Society.

Calloway, S. (2006). *Record retention periods*. . (Original work published 2001) Retrieved August 2, 2008, from HIPAA Advisory Web site: http:// www.HIPAAadvisory.com/regs/recordretention.htm

Casteel, J. (2007). *Oracle 10g SQL*. Boston: Thomson Course Technology. (Original work published 2006)

Casteel, J. (2003). *Oracle9i*. Boston: Thomson Course Technology. (Original work published 2003)

Codd, E. F. (1969). A relational model of data for large shared data banks. In E. F. Codd (Ed.), *IBM Information Retrieval* (pp. 1-11). San Jose, CA: IBM Research Laboratory.

Fowler, M., Rice, D., Foemmel, M., Hieatt, E., Mee, R., & Stafford, R. (2003). *Patterns of enterprise application architecture*. Boston: Pearson Education, Inc. (Original work published 1963)

Greenwald, R., Stackowiak, R., & Stern, J. (2004). *Oracle essentials: oracle database 10g* (3rd ed.) (D. Russell, Ed.). Sebastopol, CA: O'Reilly Media, Inc. (Original work published 1999)

Hale, L., & Levinger, J. (2003). *Oracle security guide*. Retrieved June 8, 2007, from Stanford University Web site: http://www.stanford.edu/dept/itss/docs/oracle/10g/network.101/b10773.pdf

History, overview, and implementation of HIPAA . (2008). Retrieved July 11, 2008, from UT Southwestern Medical Center Web site: http://www4.utsouthwestern.edu/HIPAA

How to: publish web application projects. (2005). Retrieved July 26, 2008, from MSN SQL2005 Developer Web site: http://msdn.microsoft.com/en-us/library/aa983453(VS.80).aspx

IEEE. (2006). *Standard for substation IED cyber security standard* (IEEE P1686). New York: Institute of Electrical and Electronics Engineers, Inc.

Jansen W., Wingoad., T., & Scarfone, K. (2008). *Computer security* (Report No. NIST Special Publication 800-28). Gaithersburg, MD: National Institute of Standards and Technology. (NIST No. 800-28)

Kamens, J. (2000). *What are the advantages/disadvantages of Kerberos vs. SSL?* Retrieved August 3, 2008, from FAQS.org Web site: http://www.faqs.org/faqs/kerberos-faq/general/section-31.html

Kiely, D. (2005). *On the horizon: improved data security in SQL server 2005*. . (Original work published 2005) Retrieved July 11, 2008, from Microsoft - technet Web site: http://technet.microsoft.com/en-us/magazine/cc160788(TechNet.10).aspx

Kuhn D., Hu., V., Polk., W., & Chang, S. (2001). *Introduction to public key technology and the federal PKI infrastructure*. Retrieved August 23, 2008, from National

Institute of Standards and Technology Web site: http://csrc.nist.gov/groups/ST/crypto_apps_infra/pki/index.html

Litwin, P. (2008). *Data security: stop SQL injection attacks before they stop you*. (Original work published 2008) Retrieved July 11, 2008, from MSDN Web site: http://msdn.microsoft.com/en-us/magazine/cc163917.aspx

Meier, J., Mackenan, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Web server security*. Retrieved August 2, 2008, from Microsoft Web site: http://msdn.microsoft.com/en-us/library/ms885809.aspx

Meier, J., Mackenan, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Improving Web application security: threats and countermeasures*. Retrieved August 27, 2008, from Microsoft Web site: http://msdn.microsoft.com/en-us/library/ms994921.aspx

McGovern, J., Ambler, S., Stevens, M., Linn, J., Sharan, V., & Jo, Elias. (2004). *A Practical guide to enterprise architecture* (M. Vincenti & M. Markham, Eds.). Upper Saddle River, NJ: Prentice Hall.

Monroe, R., Kompanek, A., Melton, R., & Garland, D. (2003). Architectural styles, design patterns, and objects. *software engineering, 1*, 241.

Padilla, R. (2005). *Don't gamble with HIPAA security compliance*. Retrieved August 2, 2008, from Techrepublic Web site: http://articles.techrepublic.com.com/5100-10878_11-5760663.htmlpart=rss&tag=feed&subj=tr

Powell, G., & McCullough, C. (2007). *Oracle 10g database Administrator Implementation & administration*. Boston: Thomson Course Technology. (Original work published 2007)

Riedl, B. (2008). Pseudonymization for improving the privacy in e-health applications. In B. Riedl, V. Grascher, S. Fenz., & T. Neubauer (Eds.), *Proceedings of the 41st Hawaii*

Sausner, R. (2008). Keeping your DBA honest. *Banking Technology News, Unknown*, 1.

Schach, S. (2007). *Object-oriented & classical software engineering* (7th ed.). New York: McGraw-Hill.

Shepherd, M., Zitner, D., & Watters, C. (2000). Medical portals: web-based access to medical information. In M. Shepherd, D. Zitner, & C. Watters (Eds.), *Proceedings of the 33rd Hawaii International Conference on System Sciences - 2000* (pp. 1-10). Halifax, Canada: IEEE.

Singhal, A. (2007). Guide to secure web services. In *Computer security subcategory, web services* (NIST Special Publication 800-95). Gaithersburg, MD: U.S. Department of Commerce. Retrieved June 1, 2008, from U.S. Department of Commerce, National Institute of Standards and Technology Web site: http://csrc.nist.gov/publications/nistpubs/800-95/SP800-95.pdf

Smid, M. (2001). *Announcing the advanced encryption standard*. Springfield, VA: National Institute of Systems Technology. (NTIS No. PB-197) Retrieved August 21, 2008, from National Institute of Systems Technology Web site: http:// csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Swanson, M., & Guttman, B. (2008). *Generally accepted principles and practices for securing information technology systems*. . (Original work published 1996) Retrieved August 2, 2008, from National Institute of Standards and Technology Web site: http://csrc.nist.gov/publications/nistpubs/800-14/800-14.pdfWatson, K., The advisory commission on consumer protection, & quality in the health care industry. (1999). Chapter 6: Confidentiality of health information. In *Consumer bill of rights and responsibilities* (pp. 16-18).. Retrieved July 19, 2008, from opm.gov Web site: http://www.opm.gov/insure/health/cbrr.htm#chpt6

Watson, K., Nagel, C., Pedersen, H., Reid, J., Skinner, M., & White, E. (2005). *Beginning visual C# 2005* (K. Mohr, T. Dinse, T. Meister, A. Smith, M. Wakefield, R. Swadley, et al, Ed.). Indianapolis, IN: Wiley Publishing, Inc.

Woody, B. (2007). SQL server security (Version 1) [Computer software and manual]. Retrieved from http://www.informit.com/guides/ content.aspx?g=sqlserver&seqNum=245>

Appendix A

Sarbanes-Oxley Act Section 302

This section is of course listed under Title III of the act, and pertains to 'Corporate Responsibility for Financial Reports'.

Summary of Section 302

Periodic statutory financial reports are to include certifications that:

- The signing officers have reviewed the report
- The report does not contain any material untrue statements or material omission or be considered misleading
- The financial statements and related information fairly present the financial condition and the results in all material respects
- The signing officers are responsible for internal controls and have evaluated these internal controls within the previous ninety days and have reported on their findings
- A list of all deficiencies in the internal controls and information on any fraud that involves employees who are involved with internal activities
- Any significant changes in internal controls or related factors that could have a negative impact on the internal controls

Organizations may not attempt to avoid these requirements by reincorporating their activities or transferring their activities outside of the United States

Appendix B

Consumer Bill of Rights and Responsibilities

Chapter Six

Confidentiality of Health Information

Statement of the Right **Consumers have the right to communicate with health care providers in confidence and to have the confidentiality of their individually identifiable health care information protected. Consumers also have the right to review and copy their own medical records and request amendments to their records.**

In order to ensure this right:

- With very few exceptions, individually identifiable health care information can be used without written consent for health purposes only, including the provision of health care, payment for services, peer review, health promotion, disease management, and quality assurance.
- In addition, disclosure of individually identifiable health care information without written consent should be permitted in very limited circumstances where there is a clear legal basis for doing so. Such reasons include: medical or health care research for which an institutional review board has determined anonymous records will not suffice, investigation of health care fraud, and public health reporting.

- To the maximum feasible extent in all situations, nonidentifiable health care information should be used unless the individual has consented to the disclosure of individually identifiable information. When disclosure is required, no greater amount of information should be disclosed than is necessary to achieve the specific purpose of the disclosure.

Rationale

The legal right to confidentiality of health care information and its essential role in the delivery of quality health care has been recognized by the United States Supreme Court, lower Federal and State courts, and Federal and State lawmakers. Similarly, a health care provider's obligation to protect the confidentiality of health information is universally recognized. The assurance that consumers' health information will remain confidential is "fundamental to effective diagnosis, treatment and healing" (Shalala, 1997).

At the same time, the quality of the health care system also depends on the regular exchange of information between providers, employers, plans, public health authorities, researchers, and other users. The changing structure of the health care system and rapid advances in information technology and medical and health care research have increased the demand for and supply of health information among traditional users such as the treating physician, and new users, such as large networks of providers, information management companies, quality and utilization review committees, and independently contracted service providers. Concerns have been raised that, under the current system of information exchange, various entities can access individually identifiable information without sufficient security safeguards and consent requirements.

Other activities undertaken to improve quality and efficiency may present new risks to the confidentiality of health information. For example, quality oversight activities by plans, providers, accreditation bodies, and regulatory agencies require detailed information about the treatment and benefit status of individual consumers. The growing role of employers in workforce health issues has also contributed to the confidentiality debate.

Congress has made repeated attempts to enact a comprehensive Federal confidentiality law but has, to date, been unsuccessful. The web of protections at the Federal and State level that has evolved in the absence of a comprehensive law leaves many aspects of health information unevenly protected. Specialized Federal protections already exist through statutes that address substance abuse, Medicaid beneficiaries, public health, research, government records, and those living with disabilities.

Several States have enacted comprehensive laws and an effort is currently under way at the National Association of Insurance Commissioners to draft a Protected Health Information Model Act for States. Other safeguards have evolved outside of the legislative arena. Accreditation bodies have incorporated requirements for confidentiality policies and patient consent (JCAHO 1996; NCQA 1997; URAC 1996) and continue to collaborate on security and confidentiality issues (JCAHO/NCQA Joint Session, 1997).

The Health Insurance Portability and Accountability Act of 1996 (HIPAA) required the Secretary of Health and Human Services to submit to the Congress detailed recommendations on: (1) the rights that an individual who is a subject of individually identifiable information should have; (2) the procedures that should be established for the

exercise of such rights; and (3) the uses and disclosures of such information that should be authorized or required (Public Law 104-191). On September 11, Health and Human Services Secretary Donna Shalala presented those proposals to the Congress (Shalala, 1997). Under the terms of HIPAA, if Congress fails to enact Federal confidentiality legislation by August 1999, the Secretary of HHS is required to promulgate regulations setting confidentiality standards.

The Secretary recommends a comprehensive Federal confidentiality law that would apply "floor preemption," meaning that the law would require that all States comply with a minimum set of confidentiality requirements but would not preempt stronger State laws.

Section 262 of HIPAA also requires the Secretary of HHS to adopt standards by February 1998 for electronic transmission of financial and administrative health care transactions (including information about claims, eligibility, payment, and injury), unique health identifiers (for individuals, employers, plans, and providers), and security.

The Commission believes that it is essential to establish a comprehensive confidentiality framework and encourages the Congress to move forward expeditiously.

Implications of the Right

- **Health plans, health providers, employers, and other group purchasers** should examine existing confidentiality protections to safeguard against improper use or release of individually identifiable information. The Commission does not intend to impede employers or providers from complying with duties established by law. Health providers, facilities, and plans should develop procedures to ensure

that when sensitive services (e.g., mental health, substance abuse, reproductive services, or treatment of sexually transmitted diseases) are involved, standard administrative techniques do not inadvertently disclose information to individuals other than the patient. This is not intended to create two standards of nondisclosure -- one for sensitive medical conditions and another for all others. It is merely a recognition that there may be high level concern about confidentiality with certain medical conditions by some patients.

Law enforcement officers, researchers, and public health agencies should examine their existing policies to ensure that they access individually identifiable information only when absolutely necessary and provide proper safeguards to assure confidentiality.

Consumers should become more aware of the content of their health records and pay particular attention to requests by providers, plans, employers, or others to gain access to those records.

URAC National Network Accreditation Standards (April 1996).

Annotated Bibliography

A guide to the Sarbanes-Oxley act. (2002). Retrieved April 28, 2008, http://www.soxlaw.com/index.htm

This is the official Web site of SOX law that provides a guide to the Sarbanes-Oxley act of 2002, covering the general rules and regulations placed into law in 2002. An additional link at this web site http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=107_cong_ bills&docid=f:h3763enr.txt.pdf links to the complete Sarbanes-Oxley Act of 2002.

Aaron, N. (2006). *Oracle database security*. Retrieved August 18, 2008, from Infosecwriters Web site: http://209.85.173.104/search?q=cache:B-QWgf08WKIJ:www.infosecwriters.com/text_resources/pdf/Oracle_NAaron.pdf+Database+View+Security&hl=en&ct=clnk&cd=27&gl=us

Aaron discusses the complexity of using user views, data encryption and various forms of security and how employees intentionally or unintentionally access information due to design and user access errors. While Aaron mentions poor database design he also reference DBA's schema errors and placing false trust in employees.

Andrews, C. (2001). *SQL server 2000 security secrets*. Retrieved July 11, 2008, from Microsoft Certified Professional Magazine Web site: http://mcpmag.com/features/article.aspeditorialsid=210

Chip Andrews discusses SQL Server 2000 and the use of certificates to validate users. Andrews writes about solving the integrated security dilemma and auditing logs. Andrews offers additional information on data encryption and the advantages of using

third-party encryption methods. Andrews also cites that there is a problem with SQL Server 2000, as encryption inside the database is not an option.

Barak, B., Herzberg, A., Naor, D., & Shai, E. (2002). *The proactive security toolkit and applications*. Retrieved June 15, 2008, from ScientificCommons Web site: http://de.scientificcommons.org/492942

The authors discuss the measures to take a proactive approach to security. They advocate cryptography but also state that security relies heavily upon using the correct architecture and integrations. The authors are concerned with breaches in security due to connectivity with legacy systems making it more difficult to use today's user ID and password standards.

Basch, R. (2008). *What is Kerberos*. . (Original work published 1990) Retrieved August 2, 2008, from Massachusetts Institute of Technology Web site: http://web.mit.edu/Kerberos/#what_is

Basch offers a brief history of Kerberos and the development of the protocol. The author's intent of this document is a pre-in to accessing the protocol versions and documentation for installation and use.

Britton, C., & Bye, P. (2004). *IT architectures and middleware* (C. Britton & P. Bye, Eds.). Boston: Pearson Education, Inc.

In this book Britton and Bye, discuss Web services and .Net technology, covering middleware, principles of distribution, application integration design and the concepts of

using a structured approach to system integration. In Chapter 6 Britton and Bye discuss the n-tier approach to application development and explain how each tier relates to each other. Britton and Bye present the various schools of thoughts regarding tier relationships.

Bunker, G. (2007, September 10). Is your privacy protected? *Hospital Management*. Retrieved May 28, 2008, http://www.hospitalmanagement.net/features/feature1311/

In this article, Bunker is concerned with medical information in electronic healthcare records (EPR) system being secure and not susceptible to unwanted audiences in a global setting. Bunker discusses the controversies of using Information Technology (IT) to maintain patient records and the security challenges of doing so. Many of the resources and information provided by Bunker come from Naomi Fulop, professor of health and health policy at King's College, London, himself, Guy Bunker the senior director of technical strategy at Symantec and Paul Cundy, spokesman for general practice computing at the British Medical Association (BMA).

Christensen, M., Schneider, G., Thayer, R., & Winters, J. (2003). S*oftware engineering* (R. Thayer & M. Dorfman, Eds.). Danvers, MA: Wiley-Interscience. (Original work published 2001)

The authors present many article on Software Engineering and the development process. This is an excellent resource for information supported by the IEEE. Articles

reference current software design in relationship with legacy systems, difficulties with software maintenance, and issues with cross platform security.

Christensen, M., & Thayer, R. (2001). *The project manager's guide to software engineering's best practices*. Piscataway, NJ: IEEE Computer Society.

In this book Christensen and Thayer, discuss the best practices in software engineering. The authors discuss all software engineering processes from the IEEE's perspective and explain the importance of following engineering standards.

Calloway, S. (2006). *Record retention periods*. . (Original work published 2001) Retrieved August 2, 2008, from HIPAA Advisory Web site: http:// www.HIPAAadvisory.com/regs/recordretention.htm

Calloway discusses the need for patient record retention that came about through HIPAA regulations. The retention periods are 2 years after a patient's death, 6 years for a patient complaint. Medicare states that a patient's record must be held for 5 years.

Casteel, J. (2007). *Oracle 10g SQL*. Boston: Thomson Course Technology. (Original work published 2006)

Casteel writes and instructs on SQL scripting within an Oracle Database. This is an excellent source for understanding SQL scripting within the 10g environment and the difference between the 10g and lower versions of PL/SQL.

Casteel, J. (2003). *Oracle9i*. Boston: Thomson Course Technology. (Original work published 2003)

Casteel writes and instructs on PL/SQL scripting within an Oracle Database. This is an excellent source for understanding using SQL as a programming language and the best practices for writing procedures, triggers and other programming functions in order to cloak database operations from the general user.

Codd, E. F. (1969). A relational model of data for large shared data banks. In E. F. Codd (Ed.), *IBM Information Retrieval* (pp. 1-11). San Jose, CA: IBM Research Laboratory.

E. F. Codd wrote this document in 1969, which he later updated, however this is the original ground breaking documentation on relational databases. Codd discusses referential integrity and cross platform data exchange with use of Structured Query Language.

Fowler, M., Rice, D., Foemmel, M., Hieatt, E., Mee, R., & Stafford, R. (2003). *Patterns of enterprise application architecture*. Boston: Pearson Education, Inc. (Original work published 1963)

Fowler discusses the patterns of n-tier programming, covering everything from domain patterns, object-relational structural patterns and distributed programming patterns. Fowler also discusses bad patterns, the cost of poor programming, and boundaries of programming.

Greenwald, R., Stackowiak, R., & Stern, J. (2004). *Oracle essentials: oracle database 10g* (3rd ed.) (D. Russell, Ed.). Sebastopol, CA: O'Reilly Media, Inc. (Original work published 1999)

The authors cover the basic information in administration and design of an Oracle 10g server and database. The authors touch lightly on security offering behavioral patters of Oracle's security methodology.

Hale, L., & Levinger, J. (2003). *Oracle security guide*. Retrieved June 8, 2007, from Stanford University Web site: http://www.stanford.edu/dept/itss/docs/oracle/10g/network.101/b10773.pdf

Hale and Levinger discuss the security within Oracle 10g. Chapter 16 is dedicated to encryption and the reasons for and against encrypting data within a table. The authors discuss the problems with sharing data encryption keys between roles. Hale and Levinger offer the algorithms that are supported in Oracle 10g. The authors also discuss the difficulty with securing a 10g database from a disgruntled Database Administrator (DBA).

History, overview, and implementation of HIPAA. (2008). Retrieved July 11, 2008, from UT Southwestern Medical Center Web site: http://www4.utsouthwestern.edu/HIPAA

The author(s) of this article explain UT Southwestern Medical Center's stance on HIPAA, the history behind incorporating HIPAA rules and regulations and the UT

Southwestern Medical Center's mission to meet the HIPAA requirements. The authors believe that UT Southwestern Medical Center personnel are meeting the HIPAA requirements. The authors also believe that the actions the IS personnel have followed seem to meet the HIPAA requirements and are within IS industry best IT practices. However, no historic base exists in which to pull information or to validate that which the IS security personnel are doing to assure information security.

Howie, J. (2002). *SQL server 2000 auditing*. Retrieved August 23, 2008, from Microsoft Web site: http://www.microsoft.com/technet/security/prodtech/sqlserver/sql2kaud.mspx

Howie wrote this paper as a guide to installing Microsoft SQL Sever 2000 and activating Server Auditing logs. Howie describes the steps involved and the consequences of server audit logs.

How to: publish web application projects. (2005). Retrieved July 26, 2008, from MSN SQL2005 Developer Web site: http://msdn.microsoft.com/en-us/library/aa983453(VS.80).aspx

This is a white paper document with no author or editor noted. Microsoft's purpose is instructional for publishing a Web application.

IBM WebSphere portal V6.0 security overview [Computer software]. (2006). Armonk, NJ: IBM.

This is a white paper document with no author or editor noted. IBM's purpose is instructional of security for an IBM Websphere server.

Jansen W., Wingoad., T., & Scarfone, K. (2008). *Computer security* (Report No. NIST Special Publication 800-28). Gaithersburg, MD: National Institute of Standards and Technology. (NIST No. 800-28)

The authors discuss various vulnerabilities of thin-client and computer security. The authors discuss a range of vulnerabilities from SQL Injection to HTML and failure to write protective code. Broken down by anatomy, source and safeguards, the authors have done an excellent presentation of thin-client security concerns, and how to counter malicious attacks.

IEEE. (2006). S*tandard for substation IED cyber security standard* (IEEE P1686). New York: Institute of Electrical and Electronics Engineers, Inc.

The IEEE published this document in 2006 as a draft and later incorporated the information as a standard. The authors of this document are presenting the best practices for user security with password protection.

Kamens, J. (2000). *What are the advantages/disadvantages of Kerberos vs. SSL?* Retrieved August 3, 2008, from FAQS.org Web site: http://www.faqs.org/faqs/kerberos-faq/general/section-31.html

The author covers the advantages and disadvantages of Kerberos and SSL. While the Kamens document is somewhat slanted and opinionated his reference to the differences between Kerberos and SSL are supported and confirmed by more reliable sources, just not as cut and dry as Kamens states them.

Kiely, D. (2005). *On the horizon: improved data security in SQL server 2005*. . (Original work published 2005) Retrieved July 11, 2008, from Microsoft - technet Web site: http://technet.microsoft.com/en-us/magazine/cc160788(TechNet.10).aspx

Kiely discusses data security by user roles, proxy accounts, and user schema specification. Kiely offers sample code for encryption of data fields.

Kuhn D., Hu., V., Polk., W., & Chang, S. (2001). *Introduction to public key technology and the federal PKI infrastructure*. Retrieved August 23, 2008, from National Institute of Standards and Technology Web site: http://csrc.nist.gov/groups/ST/crypto_apps_infra/pki/index.html

The authors wrote this document as standardization for the NIST for use of certificates, PKI's and digital signatures. The authors also discuss their interpretation of HIPAA laws and the use of certificates to meet security requirements. The document is over 54 pages of instructions and definitions. The authors provide excellent documentation that offers background and understanding of certificates, PKI's and digital signatures.

Litwin, P. (2008). *Data security: stop SQL injection attacks before they stop you*. . (Original work published 2008) Retrieved July 11, 2008, from MSDN Web site: http://msdn.microsoft.com/en-us/magazine/cc163917.aspx

Paul Litwin discusses SQL Injection attacks by explaining how SQL injection attacks work, testing for vulnerabilities, validating user input, using .NET features to prevent attacks, and the importance of handling exceptions correctly. This article is in

relationship to ASP.NET, C#, and SQL. Litwin explains how a hacker breaks into a system by injecting malformed SQL queries into a data field, and he offers sample software code to protect the database from such intrusion. Litwin also offers information on testing, evaluating errors, and unit testing.

McGovern, J., Ambler, S., Stevens, M., Linn, J., Sharan, V., & Jo, Elias. (2004). *A practical guide to enterprise architecture* (M. Vincenti & M. Markham, Eds.). Upper Saddle River, NJ: Prentice Hall.

The authors cover enterprise architecture; how the concepts and methods, patterns, and UML processes work in real life. This is an excellent source for obtaining an understanding of the enterprise architectural designs. In Chapter 11, the authors discuss data architecture from design through n-tier applications and Meta data.

Meier, J., Mackenan, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Web server security*. Retrieved August 2, 2008, from Microsoft Web site: http://msdn.microsoft.com/en-us/library/ms885809.aspx

This is an instructional documentation from Microsoft on securing a Web server. The authors cover known vulnerabilities and Microsoft's solutions to data protection. While geared directly towards Microsoft's own products, its overview on vulnerabilities relate to any Web server and connection to the Internet.

Meier, J., Mackenan, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Improving Web application security: threats and countermeasures*.

Retrieved August 27, 2008, from Microsoft Web site: http://msdn.microsoft.com/en-us/library/ms994921.aspx

The authors of this article cover some very common sense type of security measure for shoring up information presented on a Web server with thin client access. The authors also explain vulnerabilities, and countermeasures with a means of improving the user experience without jeopardizing security.

Monroe, R., Kompanek, A., Melton, R., & Garland, D. (2003). Architectural styles, design patterns, and objects. *software engineering, 1*, 241.

The authors discuss software architectural styles, design patters, and objects however, they do not advocate one approach over another. This is an excellent resource for obtaining generalized information on IEEE approved software methodologies.

Padilla, R. (2005). *Don't gamble with HIPAA security compliance*. Retrieved August 2, 2008, from Techrepublic Web site: http://articles.techrepublic.com.com/5100-10878_11-5760663.htmlpart=rss&tag=feed&subj=tr

Padilla discusses the problems with meeting HIPAA regulations and electronic employee connectivity and activity. Padilla explains the chances some institutions are taking by not being fully HIPAA compliant.

Powell, G., & McCullough, C. (2007). *Oricle 10g database administrator implementation & administration*. Boston: Thomson Course Technology. (Original work published 2007)

Powell and McCullough explain Oracle 10g database administration from the implementation of a database through its life cycle. An excellent source for administration of an Oracle database, the users, data security, troubleshooting and Oracle features.

Praities, N., & Anekwe, L. (2008, February 6). BMA brands mental health law 'unethical. *Pulsenews*, p. 10.

The author's are reporting on the British Medial Association (BMA) decision to have all mental health patients information placed into a national database. The author's note that one of the main reasons by physicians to resist a national database is the concern that information within database not being adequately secured. The author's state statistics that 9 out of 10 doctors believe that patient's records will be at risk if placed into a national depository.

Riedl, B. (2008). Pseudonymization for improving the privacy in e-Health applications. In B. Riedl, V. Grascher, S. Fenz., & T. Neubauer (Eds.), *Proceedings of the 41st Hawaii International Conference on System Sciences - 2008* (pp. 1-10). Vienna: IEEE.

The author's describe Pseudonymization; a technique where identification data is transformed, and replaced by a specifier that cannot be associated with the identification data without knowing a certain secret. The author's describe the good points of Pseudonymization and the bad points by explaining the consequences. However, the author's find Pseudonymization to be a more secure and expedient means to secure data rather than data encryption which when applied to healthcare graphics such as

mammograms can be inefficient and time-consuming. The author's also describe in detail the formulas and definition of system attributes for Pseudonymization.

Sausner, R. (2008). Keeping your DBA honest. *Banking Technology News, Unknown*, 1.

While this article relates to the banking industry, Sausner references the medical industry and the ability of DBA's to cover their tracks while snooping on celebrities medical files. This is an excellent article where Sausner discusses the use of audit logs and tracking user access in conjunction with electronic signature software that notes breaks in audit logs.

Schach, S. (2007). *Object-oriented & classical software engineering* (7th ed.). New York: McGraw-Hill.

In this book, Schachs discusses the best practices within the software development life cycle. Schach goes into detail on the Agile Methodology of software engineering through the software process, working in teams, testing and cohesion. Often he reflects on the classical software engineering process compared to OOP software engineering, and contrasting the differences.

Shepherd, M., Zitner, D., & Watters, C. (2000). Medical portals: web-based access to medical information. In M. Shepherd, D. Zitner, & C. Watters (Eds.), *Proceedings of the 33rd Hawaii International Conference on System Sciences - 2000* (pp. 1-10). Halifax, Canada: IEEE.

The author's investigate the 3-Tier based Portal for Web-based access to medical information. The author's model the process and show the complexity of security

required within the architecture. The author's define the pull, push and update process of sending and receiving data. The author's also describe the needs for data from the physician's role and the CEO's role. The references are excellent and a good choice for secondary research such as: Computer-based Patient Record Institute, Health Level 7 and the National Library of Medicine.

Singhal, A. (2007). Guide to secure web services. In C*omputer security subcategory, web services* (NIST Special Publication 800-95). Gaithersburg, MD: U.S. Department of Commerce. Retrieved June 1, 2008, from U.S. Department of Commerce, National Institute of Standards and Technology Web site: http://csrc.nist.gov/publications/nistpubs/800-95/SP800-95.pdf

Singhal describes and covers NIST guidelines for securing Web services. Section 5 is concerned with legacy software's, databases, and their security. Section 5 also covers HIPAA policy as set forth in 800-66-Rev1/Draft_SP800-66-Rev1.pdf references

Smid, M. (2001). *Announcing the advanced encryption standard*. Springfield, VA: National Institute of Systems Technology. (NTIS No. PB-197) Retrieved August 21, 2008, from National Institute of Systems Technology Web site: http:// csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Smid covers the use of AES as the new standard in encryption offering information on the history and evolution of encryption. Smid explains how AES became the standard and depicts how the functionality algorithm works.

Swanson, M., & Guttman, B. (2008). *Generally accepted principles and practices for securing information technology systems*. . (Original work published 1996)

Retrieved August 2, 2008, from National Institute of Standards and Technology Web site: http://csrc.nist.gov/publications/nistpubs/800-14/800-14.pdfWatson, K.,

The authors wrote this document as a standard supported by the NIST covering security of information and technology systems. Audit logs are covered in Chapter 3 beginning on page 50 offering a wealth of information on what audit logs should contain, how they are to be constructed and the advantage of not just creating and having them, but regular review of the information they contain.

The advisory commission on consumer protection, & quality in the health care industry. (1999). Chapter 6: Confidentiality of Health Information. In c*onsumer bill of rights and responsibilities* (pp. 16-18).. Retrieved July 19, 2008, from opm.gov Web site: http://www.opm.gov/insure/health/cbrr.htm#chpt6

This is the official government Web site on HIPAA rules and regulations. Chapter 6 is on the laws that govern health care information. The requirements and the exceptions are discussed in detail.

Watson, K., Nagel, C., Pedersen, H., Reid, J., Skinner, M., & White, E. (2005). *Beginning visual C# 2005* (K. Mohr, T. Dinse, T. Meister, A. Smith, M. Wakefield, R. Swadley, et al, Ed.). Indianapolis, IN: Wiley Publishing, Inc.

The authors of this book instruct in writing Visual C# script within the .Net framework and more specifically Visual Studio 2005. Chapter 18 covers basic Web programming while chapter 23 explains XML use within a published application. The authors warn against using XML for any object that needs security. The authors cover in

detail deploying Web application in chapter 21 and discuss data access is detail within chapter 22.

Woody, B. (2007). SQL server security (Version 1) [Computer software and manual]. Retrieved from http://www.informit.com/guides/content.aspx?g=sqlserver&seqNum=245>

Woody discusses and depicts SQL for Encrypting sensitive data and offering information on algorithms for asymmetric certification and keys. Woody also offers the warnings of how locking yourself out of your own data becomes easy, thus encryptions should not be entered into without planning.

W3C. (2008). *http://www.w3.org/Security/* (W3C security home). Cambridge, MA: Author.

This is the official web site of the W3C accepted best security practices. Also presented is additional information regarding what the W3C is meeting and working on security standards but have not yet incorporated into the W3C standards.

Glossary

AES	Advance Encryption Standard
Banner Grabbing	Connecting to remote applications and observing the output.
Blended Threat	A blended threat is a sophisticated attack that bundles some of the worst aspects of viruses, worms, Trojan horses and malicious code into one threat.
BMA	British Medical Association
Buffer Overflows	Occurs when a program or process tries to store more data in temporary data storage then it was intended to hold.
CRUD	Create, Read, Update, Delete
DES	Data Encryption Standard
Fob or Key Fob	A hardware device that randomly generates an access code that changes every 30 to 60 seconds. Combines with a PIN number the fob generated code gives access to another computer.
HIPAA	Health Insurance Portability and Accountability Act
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol – a communications protocol for the transfer of information on the Internet.
IEEE	Institute of Electrical and Electronics Engineers
Kerberos	Kerberos is a protocol designed to enable two parties to exchange private information across an open network.
IS	Information Systems
IT	Information Technology
NetBios Enumeration	To gain information about the resources available on a host or network.
NIST	National Institute of Standards and Technology
OITS	Orissa Information Technology Society
OOP	Object Oriented Programming

PL/SQL	Program Language/Structured Query Language
MVC	Model View Controller
PAC	Picture Archive and Communication Systems
PIN	Personal Identification Number
Ping Sweeps	Network scanning technique used to determine which of a range of IP addresses map to live computers.
PKI	Public Key Infrastructure
Port Scans	A series of messages sent by someone attempting to break into a computer to learn which computer network services, each associated with a *well-known* port number, the computer provides.
SOX	Sarbanes-Oxley
SQL	Structured Query Language
SQL Injection	SQL Injections is a technique that exploits a security vulnerability occurring in the database layer of an application by inserting a query into a data field.
SSL	Secure Socket Layer
SYN Flood	An attack that sends TCP connections request faster than a machine can process them causing the port to remain open and blocks authorized user(s) access.
T-SQL	Transact Structured Query Language
XML	Extensible Markup Language
W3C	World Wide Web Consortium
WWW	World Wide Web