Journal of Advanced Computer Science & Technology, 6 (2) (2017) 33-39

Journal of Advanced Computer Science & Technology

Website: www.sciencepubco.com/index.php/JACST
doi: 10.14419/jacst.v6i2.8425
Research paper

Usability meets security: a database case study

Yong Wang, Bharat S. Rawal *

IST Department, Penn State Abington, PA 19001 USA
**Corresponding author E-mail: brawal@psu.edu*

Abstract

Abstract In this paper, we review security and usability scenarios. We propose security enhancements without losing usability and apply a new approach to popular application systems. Specifically, we analyze database security for access control, auditing, authentication, encryption, integrity control, backups, separation of environment, and secure configuration. Finally, we present our recommendations for system security and usability that work together.

Keywords: *Usability; Security; Working Together; Database.*

1. Introduction

As the Internet becomes more and more popular, Internet-based network servers provide and allow information access remotely and locally. Because information can be retrieved easily, breaches of security can happen on a large scale. Popular security breaches include password exposures, unauthorized modification of data, stealing confidential documents, Structured Query Language (SQL) injections, etc. Considerable study has been performed on information security [1].

The usability and security of computer systems are not mutually inclusive. In general, the easier systems are to use, the more security problems, there are. Since the late 2000s, securing computer systems has become a major task for system administrators. In addition, for many systems, computer use is still an important factor when taking security measures.

Usable security has been studied in human-centered computing. The human-centered computing means fully understanding user and machine interactions, user behavior patterns, and work traffic to enhance system utilization. Increasingly, human-centered computing and user studies have been published in security-related journals and for security-related conferences. Even further, new conferences regarding privacy and security have emerged [2, 4].

In recent years, usable security studies have simultaneously considered knowledge from various areas. Although security research does benefit from user behavior research, this approach is limited by comparing the pros and cons of security versus system usage [12-14]. In this paper, we discuss how security can meet usability in popular application systems within a database.

The remainder of the paper is organized as follows. Section II presents related work; Section III describes our approach; Section IV provides limited electronic information access; Section V talks about checking database traffic and log files; Section VI discusses about authentication; Section VII describes different encryption scheme; Section VIII discusses database backup; Section IX discusses application security: balancing encryption and access control; Section X presents separation of environment; Section XI. secure configuration; Section XII discuss NOBE, and Section XIII presents recommendation.

2. Related work

Usability-only evaluation strategies are not helpful when determining or concentrating on security measurements. Mihajlov et al. [22] pointed out that many studies simply illustrate security and usability shortcomings [13]. Mockel [23] explained that current usability evaluation does not adequately account for the unique essence of shielded usages and different systems. Möller et al. [21] stated that the adoption of security systems is often abandoned as the best approach because of usability concerns. Most security systems fail to address usability in their design [21]. User interfaces should include the security and usability that users need from the beginning of the interface design; users need to consider functionality and system security because security tools may already be built into an application [24]. Because security is meant to protect users, when we ignore it, harm can be applied to the user or the system. A good example is medical information systems, which carry highly sensitive data [25]. Gunson et al. conducted an experimental approach on automated telephone banking systems and found that understanding of the security measures is improved when an extended authentication procedure is applied. They also detected that the cost of usability is more with a higher understanding of security. Reducing the security of a system by limiting access control increases risk [26-27]. Mihajlov et al. [28] developed a theoretical framework using quantification and quality measurements. They determined both security and usability properties in the study [1, 3, and 13]. Faily et al. [34] pointed out considerable research efforts in human-computer interaction and security in recent years. However, the majority of work pertains to studying the usability of security controls and conceptual investigation of trust and privacy. Specifically, their research focused on the needs of end users.

3. Our approach

In this paper, we review different information security methods and discuss their usability issues. Specifically, we analyze security and usability scenarios and make recommendations [3].

- Task Scenario: This scenario represents the description of the task. The context of utilization analysis discusses a wide range of approaches. These approaches describe the properties of the users, goals, and user surroundings. The context used is designed for data gathering to build useful blocks at an early stage. It also analyzes the results to see whether results are productive and fulfilling.
- Usability Scenario: This scenario discusses person behaviors while conducting a job. The usability description is domain-related to one task. The usability scenario should be discussed with types, procedures, or assessment approaches.
- Security Scenario: This scenario outlines the task scenario and includes specific security approaches. Security scenario aspects are easily seen or have no physical substance. Touchable security embraces physical facilities that control an individual approach to constructs with personal biological characteristics. Touchable security is related to electronic information or another material.

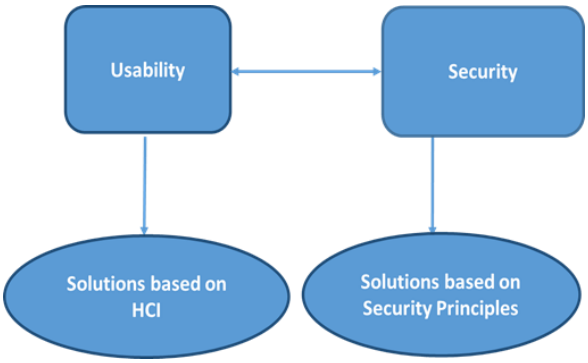

Fig 1: Usability and Security Working Together.

Anderson's rule states that naturally large databases are certainly not immune to exploitation by computer hackers. When a great system is developed for unlimited use, hackers easily access it. Moreover, if a system is designed with tight security, it often has limited usability. There is every so often a trade-off between security and usability. For database security, there are several popular approaches [10], [29]:

- Limited electronic information access
- Checking traffic data and log files (auditing)
- Authentication
- Encryption
- Data correctness checking
- Data backups
- Application security
- Separation of environments
- Secure configuration

In the following sections, we discuss security scenarios and usability scenarios. Then, we discuss whether usability and security can work together when dealing specifically with database security.

4. Limited electronic information access

Limited information retrieval is a popular security strategy implemented by organizations by disallowing logins to electronic resources. This outdated control strategy does not meet the optimal requirements in philosophy, object, subject, or implementation [9], [10], and [11].

4.1. Access control models

LaPadula and Williams presented information access and retrieval models categorized into various measurements, in general to detail-oriented operations [9], [10], and [11]:

1) Security goals—initial security requirements for the electronic resources
2) External requirements for users and the interface between security and surroundings
3) Internal needs—policies and criteria in inner parts
4) Implementation policies—how security is ensured for inner parts
5) Working properly—proposing different actions for system parts

The access control models that we present satisfy Level 3 above. The time-based access control (TBAC) and role-based access control (RBAC) models are very popular, but they only meet Level 2, which is more general.

4.2. Types of access control models

Security models can be categorized based on several factors. Research scenarios and different security policies can focus on many areas. For this study, we discuss specific security policies implemented by the model [32].

Four popular models have been developed—discretionary, mandatory access control (MAC), RBAC, and TBAC. The data owner can insert and retrieve from the first model approach. This model is a very popular one. The policies are easily defined and are discretionary because they allow users to grant other users authorized to access objects. Discretionary policies are applied in commercial systems based on their flexibility, which makes them fitted for various environments with different requirements.

MAC improves the global policy in the different security levels that are allocated to all accesses. MAC security policies govern access depending on subject and object classification in the system. Objects are passive entities storing information, for example, relations and tuples. Subjects are active entities that access the objects, normally active processes operating on behalf of users. An access control consists of two parts: a security level and a set of categories. The security level is an element of a hierarchically ordered set. Levels include top-secret (TS), secret (S), confidential (C), and unclassified (U), where $TS > S > C > U$.

RBAC: With RBAC, permissions are associated with roles. Users make a number of roles. This simplifies permission management. Roles are strongly associated with the concept of user groups in access control. A role can bring a set of users to one site and a set of permissions to another site. A user group is typically defined as a set of users only. RBAC has objects, operations, permissions, roles, users, groups, constraint sessions, and role hierarchies:

- Object—system, resource file, printer, terminal, and database
- Operation—executable image of a program that executes some function for a user
- Permission—approval to perform an operation on one or more RBAC-protected objects
- Role—a job function in an organization with some associated semantics regarding authority and responsibility conferred on the user assigned to the role
- User—a human being or a process executed by a user
- Group—a set of users
- Constraint—a relation between or among users
- Session—a map between a user and an activated subset of roles to which a user is associated
- Role hierarchy—a partial-order relationship established among roles

TBAC [35]: An access control list (ACL) is a sequential list consisting of one permit statement and one or more deny statements. Using a time-based ACL is easy and can be useful in some situations with the following steps:

1) Define time range
2) Define ACL that the time-range applies to
3) Apply ACL

It appears that object-oriented (OO) tools have inspired different methods that echo OO database management systems (DBMSs)

and updated requirements in various applications. The first approach is RBAC depending on roles. The second approach is TBAC depending on different jobs. TBAC takes categorically new ways and trials, and the usability in a specific category is well achieved.

5. Checking database traffic and log files [5]

Database auditing means observing a database to be conscious of the activities of database users. Database managers and specialists often examine databases for security purposes, for example, to ensure that those without data authorization do not have access. Azure SQL database auditing tracks database occasions and records events to an audit log in a storage account. Assess is available for Basic, Standard, and Premium service tiers. Auditing can help ensure regulatory compliance, policy enforcement, understanding of database activity, and penetration of errors and anomalies that could lead to business concerns or suspected security breaches.
In bottle line, the following activities should be audited [29], [31]:

- Administrative activity
- Login and logoff activity
- Failures
- Use of system privileges; this helps to detect intrusions
- and elevation of privilege
- Critical object access—detecting system changes that
- have not been applied, which could mean that
- a rootkit is present
- Alterations to the database structure
- Database configuration and settings checks
- Vulnerability and threats—detect vulnerabilities in the database and then monitor for users attempting to exploit them

Customized auditing is very important. When users read or revise records, it should be recorded. Fine-grained auditing can be used to monitor these kinds of activities.
Saving inspection records on a secondary system and centralizing audit records is also important. Saving records on a secondary system reduces the possibility of unauthorized activity on the audit trail. By centralizing audit records on a single system, the activity can be reviewed and monitored across databases without examining each database individually. For the Oracle database, this function is realized using an audit vault.
Auditing devices permit and facilitate adherence to agreement standards, but do not support compliance. For auditing, security does not have a significantly meaningful impact on database usages. Security and usability can work together.

6. Authentication [6], [30]

Passwords are an important authentication. Users need to input the correct password when they want to access the database. Users need to pass authentication using the information stored in the database. Passwords are created when users apply for access to the database.
Database security relies on passwords being kept secret. Passwords are vulnerable to stealing, misuse, and forgery. Several approaches can enhance password security:

- Password policy can be managed by the database administrator in user profiles.
- The database administrator has specific requirements for password complexity.
- Passwords should not be found in the dictionary. Passwords should not contain peoples' names or birthdays.
- Passwords need to be periodically changed. After some time, the database requires users to change their passwords.
- When a user has failed to log in several times, the database server automatically locks the account.

Strong authentication must be included in network security. Strong authentication methods are the client to server and server to server. These methods embrace Kerberos, Remote Authentication Dial-In User Service (RADIUS), token cards, smart cards, Distributed Computing Environment (DCE), biometrics, Public Key Infrastructure (PKI), and certificate-based authentication.
Kerberos and CyberSafe: Kerberos is a third-party authentication system developed by the Massachusetts Institute of Technology. Kerberos depends on shared secrets. It assumes that a third party is secure. The third party supplies single sign-on capabilities, centralized password storage, and database link authentication. It does through a Kerberos verification server.
Kerberos single sign-on has several benefits. With one centralized password store, it reduces administrative overhead and requires users only to remember one password. It controls network access time. By using the Data Encryption Standard (DES) and CRC-32 integrity, it improves system security against unauthorized access and packet relay.
RADIUS: RADIUS is an industry protocol applied by authentication vendors. It has the functions of user authentication, authorization, and accounting between a client and authentication server. Organizations have implemented it such that users can access the network remotely. Enterprises have established it as a standard because of flexibility and the capability to centralize all user information for reduced cost in user administration.
Token cards: Token cards are easy to use because of several mechanisms. Some cards have one-time passwords that are synchronized with an authentication service. The server can verify the password provided by the token card by communicating with the authentication service. Other token card has a keypad and operates on a challenge-response basis. In this way, the server offers a challenge that the user enters into the token card. The token card has several advantages including strong authentication, ease of use, ease of password verification, and enhanced accountability.
Smart cards: A smart card is a hardware device like a credit card with memory and a processor. The card is read by smart card readers located at the client workstation. The smart card has several benefits including increased security (depending on two-factor authentication), improved performance (with hardware-based encryption chips), accessibility from any workstation, and memory.
DCE: DCE is produced by an open-source software foundation. It is a group of integrated network services that operate in multiple systems. The network services embrace remote procedure calls, directory service, threads, security, distributed file service, distributed time service, and diskless support. DCE is middleware between distributed applications, operating systems, and network services. DCE is implemented using a client-server model. Using the services and tools from DCE, users can create, use, and maintain distributed applications across a heterogeneous environment.
Biometrics: Biometrics is a recently introduced way to achieve strong authentication. Basically, it uses physical characteristics of users to authenticate access, such as fingerprints or voices.
PKI and certificate-based authentication: PKI is an industry-standard set of procedures and rules that are used to provide secure information exchange. It can provide encryption and access control. It also provides secure credentials with digital certificates that are used to authenticate all users.
Though security and usability are two different attributes in database systems, they are dependent on each other because of login names and password checks. Previous research has found that users have difficulty memorizing randomly generated passwords. An experiment indicated that 12% of users are capable of remembering their passwords two months after creating them. On the other hand, passwords that rely on mnemonic phrases are easier to remember than randomly generated passwords. These two approaches appear to have a strictly comparable security level, so by approaching user education with mnemonic passwords, we obtain obvious enhancement in security.
Various types of dependency exist between usability and security [12-14]. Sometimes, the more usable approach can result in more secure systems if the system applicants do not make incorrect judgments.

7. Encryption [7]

By encrypting [19], we mean that a process converts data in a database into "cipher materials." Cipher data are not readable without applicant decryption. The goal of database encryption is to ensure that data stored in a database are kept from being retrieved by hackers—to prevent illegitimate access and users with malintent. Data encryption can improve database security because the database is not readable or understandable for hackers. Various algorithms and approaches are used to encrypt databases; we will discuss some of the most popular in the following paragraphs [20].

7.1. Transparent/external database encryption

The transparent data encryption (TDE) technique is applied for encrypting an entire database. TDE requires silent data encryption. Silent data can be described as inactive data. The data are not being utilized or traveling over a communication medium at any time. For example, a Microsoft® Word® document saved on any computer is called silent before a user clicks on and revises the document. Generally, silent data are kept in primary or secondary storage media such as compact disks (CDs), tapes, and hard disks. Security and theft always become an issue when holding a significant amount of sensitive data on physical storage media. TDE prevents these data from being accessed and interpreted by unauthorized users with malicious intentions. If a user cannot read the data, the information is rendered useless, thus reducing the incentive for theft.

7.2. Column-level encryption

To understand column data encryption, it helps to understand database tables and design schemes. A conventional relational database is made of tables. Every table has rows and columns to store data. While TDE normally encrypts the whole database, column-level encryption provides options for column data to be encrypted in the tables.
Column-level encryption has both strengths and weaknesses as compared to encrypting a whole database. Column-level encryption allows more flexibility compared to the encryption of an entire database system. Moreover, there can be a distinct and individual encryption key for every column inside a digital data system, which emphatically adds a level of difficulty for creating spectrum tables and thus reduces the probability of losing or leaking stored data within each column. Speed is the only limiting factor associated with the column-level database encryption technique.

7.3. Field-level encryption

Empirical work has been completed on implementing database transactions (similar to mathematical procedures) for encrypted fields without applying decryption. Robust encryption requires the arbitrary mechanism of decryption. This technique is not as good as the randomized one.

7.4. Encrypting file system

It should be kept in mind that conventional database encryption methods typically encode and decrypt the data of a database scheme. These databases are maintained by DBMSs that operate on top of the host operating system [19]. This reveals possible security flaws because the encrypted database may be consecutively run on a reachable and possibly susceptible operating system like Windows® XP. The extent of encryption for the Encrypting File System (EFS) is much wider because it can encrypt information that is not a portion of a database system, whereas TDE can only encrypt the database files. Although EFS does broaden the scope of encryption, it also reduces database performance. In addition, it can be management questions, as system managers need interaction with the operating system using EFS. Because of performance issues, EFS is not typically used for database applications that require various database inputs and outputs.

7.5. Symmetric and asymmetric database encryption

a) Symmetric database encryption

In symmetric encryption, database encryption requires a unique private key be used for data that are saved and requested from given a database. This private key modifies information in a fashion that makes it indecipherable without first performing decryption. Data are encrypted when placed and decrypted when given to the end user with the private key. When sharing information through the database, the acquiring individual must have a copy of the secret key to decrypt the data. If the information is shared through a database, the end user needs a transcript of the secret key used by the sender to decrypt and read the data. Symmetric encryption has a potential problem in that relevant information can be leaked if the private key is advertised to individuals who should not have access to the data.

b) Asymmetric database encryption

Asymmetric encryption involves two dissimilar types of keys for encryption: private and public keys. A public key can be retrieved by anybody and is distinctive to a particular user, while a private key is an undisclosed cryptographic key that is exclusive and only identified by one user. In general, the public key is used for encryption while the private key used for decryption. For instance, if Person A wants to send a message to Person B using asymmetric encryption, Person A would encrypt the information using Person B's public key and then send the encrypted message. Person B would then be capable of decrypting the information with the help of the private key. Person C would not be capable of decrypting Person A's message because Person C's private key is not the same as Person B's private key. Asymmetric encryption is widely practiced and is the most secure approach.

7.6. Database integrity

The state of a database in which all data values are exact means that it (a) reflects the state of the real world given the limitations of accuracy and timeliness and (b) obeys the rules of consistency. The maintenance of the integrity of the database involves verifying the integrity and recovery of the error state that can be detected. Normally, this is the responsibility of the DBMS database administrator.

7.7. Data Integrity

Data integrity is about data correctness in the database. There are two kinds of integrity: data accuracy and obeying rules with mutual consistency. The maintenance of database integrity involves integrity-checking and data recovery from any incorrect states. Data integrity is the responsibility of the database administrator [8].
The term "data integrity" has different meanings for different people, but the most difficult problem for organizations is the semantic integrity of the data. Because the database owners are storing and retrieving data from various sources, the data integrity characteristic of accuracy is sometimes ignored. Maintaining data integrity requires proper design, processes that meet business objectives, and constant vigilance.
Semantic data integrity requires fully understanding the meaning of database design and data transactions among different types of data. The DBMS provides choices, controls, and methods to meet the semantic integrity of the data stored in its databases.

7.8. Check constraints supported in all different databases

Check constraints are available in all of the major DBMS products, including MySQL, DB2, Oracle, and Microsoft SQL. Neverthe-

less, they are frequently ignored in designed databases. Check constraints can enhance data integrity without requiring procedural approaches. One restriction is that data that can be stored in a column in different database tables.

Review restrictions place specific data value constraints on the contents of a column using Boolean expressions. In the database, different operations can be used to manipulate data. These include Where, Insert, and Update. Aggregation can also be performed for the database data. Any trials to modify the column data will cause the expression to be constraint-checked. If the replacement corresponds to the Boolean expression, the update operation is authorized to continue. Otherwise, the operation is considered a constraint violation.

Check constraints are easy to check because they are written using standard SQL statements. There are two constraints: a restriction name and a check condition. The SQL identifier defines the limitation name. The test condition determines the actual constraint logic operation. It can be determined using all logic qualifiers (>, <, =, <>, <=, >=).

8. Database backup

A full database backup stores all data in storage media such as CDs and tapes. The backup includes a transaction log and all database tables. When all backups are installed, the full database can be recovered. The primary advantage of a full backup is that full copies of data are available in a single medium. This results in minimal time to restore data. The disadvantage is that it takes a longer time to conduct a full backup than other types of backups [30], [33].

Full backups are run periodically. For data centers that have a small amount of data, full backups are run daily or even more frequently. Typically, full back up operations is conducted to combine with either incremental or differential backup.

Incremental backups copy only the data that have changed since the last backup operation. The revised time stamp is used and compared to the time stamp in the last backup. Modified files are tracked by recording the date and time of backup operations.

Because an incremental backup only copies data changed since the last backup, it may be run as often as desired. Only the most recently changed data are stored. The advantage of this approach is that it copies a smaller amount than the full-scale backup. These operations complete faster and need fewer media to store the backup data.

A differential backup operation is similar to an incremental backup in that it copies all data changed since the previous backup. However, it continues to copy all data changed since the previous full backup. It stores more data than an incremental backup on subsequent operations, but typically less than a full backup. Differential backups need more space and time to complete than incremental backups, though less than a full backup.

From these three primary backup methods, it is possible to develop an approach to protect data. In general, one of the following backup approaches is applied:

- Full daily
- Full weekly + differential daily
- Full weekly + incremental daily

When a database is large in size, full database backups take a long time to finish. An extensive database needs more storage space, so it may supplement a full database backup using differential backups. In the database recovery model, after each backup, the database has potential data loss if a disaster were to happen. The data loss exposure increases with each update until the next backup, when the data loss exposure nets to zero and a new cycle of work-loss exposure starts. For databases that use full and logged recovery, database backups are required but not sufficient. Transaction log backups are also expected.

Database backup has an additional benefit of security. When database security is compromised, the backup can be restored to provide data availability and integrity for the database. As a result, database backups and security can be incorporated together.

9. Application security: balancing encryption and access control

In many instances, the most sensitive data in a company are saved in databases. Medical records, credit card numbers, employee records, social security numbers, and other such data are subject to privacy regulations and must be protected. It is important that we understand application security scenarios surrounding SQL injection, privilege elevation, authentication, etc. When we implement more security layers, accessibility is reduced. However, security must be evaluated with the need to access data for genuine business use such as backups and remote replication for business continuity. Encryption is the most powerful tool for the protection of personal data, but it must be applied carefully to balance security and business interruption. There are some best practices for database security and control by establishing an encryption/access control balance.

9.1. Quality standards for security evaluation

Ugochi Oluwatosin et al. conducted research on the usability and security user interface design described in Table 1 [13].

Table 1: Various Quality Standards for Security Evaluation [13]

Security Criterion	Description
Revelation	Insufficiency of the verification password is hinged on aspects of a system and its users
Secrecy	Authentication password certainty relies on system and human factors
Privacy	Protecting user's personal information from remaining endangered
Breakability	Weakness of systems authentication
Abundance	Quality of accessible authentication passwords

Revelation: Revelation takes the access level of a secret authentication code from a user and system perspective. There are several methods that might disclose a verification key. One of these methods deals with frequent pop-up warnings. Because of frustration with such notices, the operator might approve the release of data that should have been kept secret [14].

Secrecy: The capability to forecast a verification key is thought to be complex work. The effort of thoughtful randomness comes from the reduced awareness of randomness that those users might present. To decide whether the authentication being designated is arbitrary, it is vital to consider clarity, the absence of association with previous or following words, and uniform distribution. For the same probability of distribution over the entire argument and exclusivity, it is a catastrophe to produce a similar order of words casually. The key feature of this security standard is to anticipate and detect how an individual can discover a password [15].

Privacy: Privacy denotes the number of individual information pieces essential for the verification part of the method [16]. A compromised password can break security and confidentiality and can cause a user identity to be stolen. Defining whom to trust with personal information is a major challenge. Unfortunately, most users do not have abilities in risk calculation, specifically where privacy choices are concerned.

Breakability: This is about the strength exerted by an invader to circumnavigate security features of a structure. The attackers have access to either the system or the codes that produce the verification password. Depending on how the scheme is designed, the invader may practice one of four approaches to detect the user's key: keylogging, research, dictionary, or brute force [17].

Abundance: This condition computes the password verification space. In other words, some likely keywords can be used to produce a password. To generate new passwords, some common passwords counterbalanced by some uncommon passwords are

frequently used in daily practice. This approach has a clear influence on the infiltration level of the verification password. This method increases complexity for hackers when attempting to compromise the targeted system [18].

10. Separation of environment [29]

Separation of production, testing, and development or similar environments has been an information security best practice for many years. This approach has used different requirements, standards, and audit trails. The environments and the data in storage need to be separate. Developers and their environment pose a significant risk to the organization with the access they hold. The designers only have access to the development environment, but there is a need for more access rights. If access does not need to be given, developers should be given read-only access with minimal privileges. Their access needs to be audited.

Production data should not be dispersed to other environments. These include testing and other environments. The data can be transferred to other environments without being compromised. For instance, Oracle data masking can protect data identification. Data masking can protect sensitive information like social security numbers or credit card numbers. These sensitive data are replaced with realistic but simulated values. In this way, referential integrity is maintained, and the applications continue to work. These fully functional masked databases are safe to transfer to nonproduction surroundings. Even if hackers gain access to these environments, the data are not useful.

In general, separation of production, testing, and development is a very good approach to protect database security. Separation of environment does not reduce database usability. It increases ease of use for production, testing, and development. Usability can actually be increased while security is enhanced—they can work together.

11. Secure configuration

We are responsible for securing application systems in a given environment. As mentioned previously, many security practitioners do not understand database security. The challenge in a database is that the complexity is overwhelming, which can lead to mistakes.

To address database security, many tools are required to deploy a secure configuration lifecycle. These include security scanning, database detection, configurations, and error remediation. Thinking about security needs to consider the database and its environments, including used operating systems and applications. Following are key areas that should be given attention for protecting database security.

- Default accounts: The Oracle database install several default user accounts. Once successful installation of the database occurs, the database configuration assistant automatically locks and expires most default database user accounts. If a manual installation is performed, no default database users are locked upon successful installation.
- Users and behaviors: Different users may have different behaviors when they access the database.
- Password compromise: User passwords are made public. Patching application: When vulnerabilities are discovered, a patch needs to be applied to secure the system.
- Access privileges: Necessary privileges only should be granted. Database users or roles should not be provided more privileges than necessary. In other words, the least amount of privileges is given for users to perform their jobs effectively.
- Parameter settings: There is a general idea of how to secure database systems.
- Password management: There is a set of policies to create or destroy passwords.
- Profiles: These are general information about database environment settings.
- Auditing: Sensitive information, SQL statements and privileges, database traffic and activity, and settings should be audited.
- Securing the network: Security for network communication is improved by using client and network guidelines to ensure protection. Using the Secure Socket Layer (SSL) is essential to enable the best security for authentication and communication.

1) Securing the client connection: Because authenticating clients is problematic, user authentication is performed. This method avoids client system issues that contain false Internet protocol (IP) addresses, hacked operating systems, and stolen client system identifiers. Three approaches include enforcing access control and authenticating clients, configuring the connection using encryption, and setting up string authentications.
2) 2. Securing the network connection: Protecting the network and its traffic from inappropriate access, or modification is essential for network security. All paths of data travel should be considered.

Database vendors have begun seriously thinking about security. Solutions such as the Oracle Configuration Management Pack are used to monitor and identify configuration vulnerability. Once a database is secure, it needs to be monitored for configuration vulnerabilities with future changes. Specifically, intentioned users may change system configurations, and the system becomes vulnerable again.

It is imperative that configuration changes are continuously watched. Many regulations require the system to be configured in a specific state. By performing long-term monitoring of modifications, management can be informed when a database does not meet organizational security and regulatory requirements.

Secure configurations do not have a specific impact on usability. Some configurations are mandatory for system security. Without secure configuration, it is difficult to build a secure system. In general, we can build a useful and secure system at the same time.

12. Nth-order binary encoding [36]

Nth-order binary encoding (NOBE) allows recursive encoding and a data compression technique that offers a much better compression ratio than the best available compression technique. The huge binary string is transmitted with the help of a few elements (b, c, n) of series functions. The detailed process was described by [36], [37].

Example

1101010101010101010101010111100001111010101011111110001101010101111011...

Is converted to

$$F(b, c, n) = c0\,(b^{0}) + c1(b^{n}) + c2\,(b^{nn}) + c3\,(b^{nnn}) + \ldots + cn\,(b^{nnnnnnnnnnnn\ldots n}) \quad (1)$$

The combination of the proposed encoding technique and Transport Layer Security (TLS/SSL) enhances the general security of data being transmitted between Web browsers and servers (HTTPS) [36]. Combining various elements of function F (b, c, n), such as base (b), coefficient (c), and power (n) within a Secure Shell (SSH) tunnel adds extra security to the data on the wire [36].

13. Recommendations

The key concern is whether security and usability balances are required. A pervasive opinion is that usability is forgone to realize meaningful improvements in security. We intend to achieve usability gains without forfeiting security. We must go further than accepting human-centered ethics and include individual decision-making.

Individuals do not want to accomplish security structures actively. Furthermore, advanced operators must be able to more easily and more efficiently comprehend and manage security structures. This puts a burden on system designers to select decision requirements given to users.

In the future, usable security may mean changing command relations between users and systems to maintain human-centered interactions. Thus, we only require high system inputs that include user plans and intentions.

To be able to construct reliable, efficient, and usable security schemes, we must have specific strategies that account for specific constraints of the usability domain and their potential consequences for security

References

[1] Serge Malenkovich, 2012, Usability and Security: The endless pursuit of perfection. Security Comments.

[2] Ronald Kainda, Ivan Flechais, and A. W. Roscoe, 2010, Security and Usability: Analysis and Evaluation.

[3] Christina Braz, Ahmed Seffah, and David M'Raihi, 2007, Designing a trade-off between usability and security: A metrics based-model, International Federation for Information Processing, pp114-126.

[4] Lorrie Faith Cranor, and Norbou Buchler, 2014, Better Together: Usability and Security Go Hand in Hand, IEEE computer security, and privacy. November/ December 2014, pp. 89-93.

[5] Get started with SQL database auditing, https://azure.microsoft.com/en-us/documentation/.../sql-database-auditing-get-started.

[6] Miloslav Hub, Jan Capek, Renata Myskova, Relationship between security and usability-authentication case study. International Journal of computers and communications, pp1-9.

[7] Database Encryption, https://en.wikipedia.org/wiki/Database_encryption (access the website on August 10, 2016).

[8] Improving Data Integrity Using Check Constraints, http://www.dbta.com/Columns/DBA-Corner/Improving-Data-Integrity-Using-Check-Constraints-99795.aspx.

[9] Daniel Cvrcek, 1998. Access Control in Database Management Systems. http://www.fit.vutbr.cz/~cvrcek/confers98/datasem/datasem.html.cz .

[10] Ross Anderson, 2005 Security Engineering, 2nd edition, Wiley Publisher.

[11] Database Application Security: Balancing Encryption and Access Control. 2016, http://searchsecurity.techtarget.com/tip/Database-application-security-Balancing-encryption-access-control.

[12] Eugene Schultz, Robert Proctor, Mei-ching Lien, Gavriel, Salvendy, 2001, Usability and Security: An Appraisal of Usability Issues in information security Methods. Computers and Security, Vol 20, No 7., pp620-634. https://doi.org/10.1016/S0167-4048(01)00712-X.

[13] Nwokedi, Ugochi Oluwatosin, Beverly Amunga Onyimbo, and Babak Bashari Rad. "Usability and Security in User Interface Design: A Systematic Literature Review." International Journal of Information Technology and Computer Science (IJITCS) 8, no. 5 (2016): 72. https://doi.org/10.5815/ijitcs.2016.05.08.

[14] R. Dhamija and L. Dusseault, "The seven flaws of identity management: Usability and security challenges," Security & Privacy, IEEE, vol. 6, pp. 24-29, 2008. https://doi.org/10.1109/MSP.2008.49.

[15] P. N. Son and H. Y. Kong, "An Integration of Source and Jammer for a Decode-and-Forward Two-way Scheme Under Physical Layer Security," Wireless Personal Communications, vol. 79, pp. 1741-1764, 2014. https://doi.org/10.1007/s11277-014-1956-z.

[16] U. Habiba, R. Masood, M. A. Shibli, and M. A. Niazi, "Cloud identity management security issues & solutions: taxonomy," Complex Adaptive Systems Modeling, vol. 2, pp. 1-37, 2014 https://doi.org/10.1186/s40294-014-0005-9.

[17] K. Renaud, Evaluating authentication mechanisms, in Security and Usability: Designing Secure Systems That People Can Use, L. Cranor and S. Garfinkel, Editors. 2005, O'Reilly Media: Stebastopol, C.A. p. 103-128.

[18] P. Mayer, M. Volkamer, and M. Kauer, Authentication Schemes - Comparison and Effective Password Spaces in Information Security, A. Prakash and R. Shyamasundar, Editors. 2014 Springer International Publishing: Hyderabad, India. p. 204-225.

[19] Carlos Cid, Sean Murphy and Matthew Robshaw,2004 "Computational and Algebraic aspects of the Advanced Encryption Standard," In Proceedings of the Seventh International Workshop on Computer Algebra in Scientific Computing.

[20] Yuan Kun, Zhang Han Li Zhaohui, 2009 "An Amended AES algorithm predicated on chaos," Multimedia Information Networking and Security,INES'09 International Conference.

[21] S. Möller, N. Ben-Asher, K.-P. Engelbrecht, R. Englert, and J. Meyer, "Modeling the behavior of users who are confronted with security mechanisms," Computers & Security, vol. 30, pp. 242-256, 2011. https://doi.org/10.1016/j.cose.2011.01.001.

[22] M. Mihajlov, B. J. Blažič, and S. Josimovski, "Quantifying Usability and Security in Authentication," vol. pp. 626-629, 2011. https://doi.org/10.1109/COMPSAC.2011.87.

[23] C. Möckel, "Usability and Security in EU E-Banking Systems-Towards an Integrated Evaluation Framework," vol. pp. 230-233, 2011. https://doi.org/10.1109/SAINT.2011.42.

[24] M. Bourimi, R. Tesoriero, P. G. Villanueva, F. Karatas, and P. Schwarte, "Privacy and security in the multi-modal user interface modeling for social media," vol. pp. 1364- 1371, 2011. https://doi.org/10.1109/PASSAT/SocialCom.2011.49.

[25] M. Minami, K. Suzaki, and T. Okumura, "Security considered harmful a case study of tradeoff between security and usability," vol. pp. 523-524, 2011. https://doi.org/10.1109/CCNC.2011.5766529.

[26] N. Gunson, D. Marshall, H. Morton, and M. Jack, "User perceptions of security and usability of single-factor and two-factor authentication in automated telephone banking," Computers & Security, vol. 30, pp. 208-220, 2011. https://doi.org/10.1016/j.cose.2010.12.001.

[27] M. Mihajlov, B. Jerman-Blazic, and S. Josimovski, "A conceptual framework for evaluating usable security in authentication mechanisms-usability perspectives," pp. 332-336, 2011. https://doi.org/10.1109/ICNSS.2011.6060025.

[28] S. Chiasson, A. Forget, R. Biddle, and P. C. Van Oorschot, "User interface design affects security: Patterns in clickbased graphical passwords," International Journal of Information Security, vol. 8, pp. 387-398, 2009. https://doi.org/10.1007/s10207-009-0080-7.

[29] White Paper, "Make database security an IT security priority," https://www.san.org/raeding-room/whitepapers

[30] Oracle Security Overview, "Authenticating users to the database," http://docs.oracle.com/cd/B12037_01/b10777/authuser.htm.

[31] R. Barnes, 2010, "Database auditing: Best Practices," http://www.isaca.org/chapter1.

[32] Indu Kashyap Kriti, "Database security& access control models: A brief overview," International Journal of Engineering Research & Technology, Vol. 2, May 2013, pp743-751.

[33] White paper, "Database security guide," http://docs.oracle.com/cd/B28359_01/network.111/b28531/guidelines.htm

[34] S. Faily, J. Lyle, and A. Simpson, "Usability and security by design: A case study in research and development," https://doi.org/10.14722/usec.2015.23012.

[35] CiscoZine, "Time-based access lists," http://www.cisocozine.com/time-based-access-lists.

[36] Bharat S. Rawal, Songjie Liang, Shiva Gautam, Harsha K. Kalutarage, and PandiVijayakumar, "Nth Order Binary Encoding with Split-protocol," International Journal of Rough Sets and Data Analysis (IJRSDA). In press.

[37] Bharat S. Rawal, Harsha K. Kalutarage, S. Sree Vivek and Kamlendu Pandey, "The Disintegration Protocol: An Ultimate Technique for Cloud Data Security," 2016 IEEE International Conference on Smart Cloud, in press. https://doi.org/10.1109/SmartCloud.2016.9.