A Novel SHA-1 approach in Database Security

Aarthi.G
Research Scholar
Mother Therasa Women University
Kodaikanal.

Dr. E. Ramaraj
Director , Computer Centre
Alagappa University,
Kaaraikudi.

Abstract:

Database security is a major research area and includes topics such as database security, intrusion detection, DDos, privacy preserving ,data mining and related papers in designing information systems that protect the privacy and ownership of individual or company information . A database contains data ranging from different degree of confidentiality and is globally accessed by variety of clients. As the importance of the database become more and more essential in business, database security turn up to isanimportant issue in order to protect data from its vulnerability to potential attackers and cryptanalysts. Encryption& decryption adds an additional layer of security to make data unusable if someone gets unauthorized access to the raw data. The new innovate proposed technique is going to design the database encryption with high performance. Database encryption is being shown as the strongest security alternative for the data protection. The main objective of thispaper is to give data security protection at data rest in database.

Keywords:

Data security, Data Rest, Encryption, SHA-1, Digest.

Introduction:

Important business and valuable data in the databases is an obvioustarget for hackers. Although access control hasbeen deployed as a security mechanism almost sinceinitiation of large database systems, so farsecurity of a DB was considered as an additionalproblem to be addressed when the need occur andafter threats to the secrecy and integrity of data hadoccurred. Nowadays many major database companiesare adopting the loose coupling approach and addingoptional security support to their database. You can use the encryption features of your DatabaseManagement System or performencryption and decryption inside &outside the database.

Each of these approaches has its advantages anddisadvantages. The enhancing method of adding securitysupport as an optional feature is not verysatisfactory, since it would always increase thesystem performance and more importantly, it ismakes to open new security holes to attackers

Encryption Process:

The need for strong data encryption is growing rapidly in data breaches and data stealing reported nationwide. The Identity Theft Resource Center recently reported a 47 percent increase in the number of data breaches in 2008 compared with the previous years.

In response, regulators have raised the stakes with Massachusetts and Nevada recently passing tough new data protection laws for businesses mandating encryption of all personal information stored on laptops.To protect against mounting business threats, C-level executives, security officers and IT managers must stop sensitive corporate data from leaking outside their organizational walls. Their search often ends with aftermarket encryption software, a far from ideal answer. Such solutions are vulnerable to attack, cause adverse effects on employee productivity and are complex to install within an enterprise. In the end, these hurdles have led software encryption to fall short of providing the protection and return-on-investment that organizations require to meet both their business and compliance goals.

Only a few database brands are currentlysupporting a row or the page level encryption that cost leads over largerdata. The third method is the location of theencryption service local service, remote procedure service or network attachedservice.

There are three main methods to encryption support in databases at rest of data. One is the integrity of data to be encrypted or decrypted. The field, the row and the page,are the alternatives. The field is the best choice because it wouldminimize the number of bytes encrypted. However, as we have discovered, thiswill require methods of embedding encryption within relational databases ordatabase servers. The second method is software versus hardware levelimplementation of encryption procedures. Our results show that the different choice makessignificant impact on the performance of data access and transaction. We have discovered encryption withinrelational databases based on hardware level implementation of encryptionalgorithms entail a significant startup cost for an encryption operation. Eachmodel also gives different performance results, tuning possibilities andencryption capabilities.

Selecting the moment of implementation not only exhibit the work thatneeds to be done from an integration perspective but also significantly affects theoverall security model. The sooner the encryption of data occurs, the more securethe environment however, due to distributed system logic in application anddatabase environments, it is not always practical to encrypt data as soon as itenters the network. Encryption performed by the DBMS can protect data at restbut you must decide if you also require protection for data while it's movingbetween the applications to the database.

Database encryption[11] allows enterprises to secure data at rest as it is written to andread from a database. This type of development is typically done at the columnlevel within a database table and if tied with database security and accesscontrols, can prevent theft of critical data. Databaselevel encryption protects thedata within the DBMS leveland also protects against a large range of threats and vulnerability,including storage media theft, well known storage attacks, databaselevel attacksand malicious database administrator. DB level encryption eliminates all applicationchanges required in the application-level model and also addresses a growingtrend towards embedding business logic within a DBMS through the use ofstored procedures and triggers. Since the encryption &decryption only occurswithin the database, this solution does not require an enterprise to understand ordiscover the access characteristics of applications to the data that is encrypted.While this solution can certainly secure data, it does require some integrationwork at the database level, including modifications of existing database schemasand the use of triggers and stored procedures to undertake encrypt and decryptfunctions. Additionally, careful consideration has to be given to the performanceimpact of implementing a database encryption solution, particularly if supportfor accelerated index-search on encrypted data is not used. First, enterprises mustadopt an approach to encrypting only sensitive fields. Second, this level ofencryption must consider leveraging hardware to increase the level of securityand potentially to offload the cryptographic process in order to minimize anyperformance impact. The primary vulnerability of this type of encryption is thatit does not protect against application-level attacks as the encryption function isstrictly implemented within the DBMS.

Proposed Method:

If encryption features are available within your DBMSproduct, you can use data cryptology within thedatabase at rest and the process will be transparent to your front endapplications. The data is converted as digest as soon as it is stored inthe database. Any data that comes or leaves the database though will be transported as clear text. This is one of thesimplest database encryption strategies but it presentsperformance ratio and security considerations that mustbe evaluated.

Encryption generally is implemented within the databasethrough a database procedure call. Some vendors support limited encryptioncapabilities through database add-ons. Other vendors mayonly provide all or nothing support for encryption eithertheentire database is encrypted or nothing is. While thismay make sense for protecting your backup copies,encryption of the entire database means additionalprocessing is expended on non-sensitive data an overkillsituation resulting in unnecessary performance degradation.A major drawback to encrypting data inside the database is theextra processing load. Because encryption and decryption areperformed within the database, the DBMS is asked toperform additional processing not only when the data isstored, but each time it is accessed. This additional processingcan leads performance degradation.

Providing security using procedure call is one of the methods. Theprocedure has to locate the stored encryption key and query it and also DBMS must verify the procedurecan access the key. The database procedure then uses the keyin the encryption algorithm and returns the encrypted result.Reading the data requires the same procedure in reverse.

For example, an application that does a sortedreport based on customer information data and accesses a databasecontaining digestcustomer id. The database procedurefor decrypting an item is executed against each encrypteddata item. If it's a large report, that can add up to a lot ofextra processing. On the other hand, applications thatdepend on indexes built on encrypted data make the processeven slower. For performance it is advisable to architect thedata so that encrypted data is not indexed but if you mustencrypt indexed data, encrypt the search value beforeperforming the search. This means that the search proceduremust be changed, and will require access to the encryptionfunction as well as the encryption key.

In our research we are storing the key value inside the database itself like maintain the separate table for the key values. Key values generated by random algorithms. In our approach we have chosen secure hash algorithm (SHA-1) for random digest generation. In common SHA applied for text but we can also apply for numbers. If we are applied the encryption to all the fields present in the database, it increases the complexity of data processing in the aspect of fetching , retrieve , storing, etc. whenever we access the database encryption and decryption process will happen for every field.

To overcome this complexity we will go to do in the single field that is act as primary part of the row or column. In simple we can also implement in the primary key field.Hash functions sometimes called 'digest' are a kind of method for a text or a data file. SHA1 creates an almost-unique 160 bit signature for a text.SHA-1 is an eminent method of the most secure hash algorithms. It is used in SSL, Pretty Good Privacy, XML Signatures. It is defined in the National Institute of Standards (NIST) and Technology standard 'FIPS 180-2'. NIST also provide a number of test factors to verify correctness of implementation.

A hash is not encryption method. It cannot be decrypted back once encrypted to the original text this makes it suitable when it is appropriate to compare 'hashed' versions of texts, as opposed to decrypting the text to obtain the original version. We couldn't decrypt the digest then how can we apply in our database? There lies the concept if we implement the encryption and decryption scheme for all the fields it leads in veindue to that we selected the single field which element should be converted as digest using SHA-1 algorithm that should be stored in the database. Whenever query coming to access the database corresponding field (primary field) should be converted in to digest that digest will matched with the existing database element and retrieves the data element but we have to maintain the separate data file for primary id with the digest for this case. We can also use this digest as index to access this fields. One thing it's obvious thing surely it will take extra memory space to store and handle the transaction.

SHA-1 applications inside the database aresuch as stored passwords, challenge handshake authenticationand digital signatures.

- Within the database to validate a password, you can store a hash of the password, then when the password is to be authenticated, you hash the password the user supplies and if the hashed versions match, the password is authenticated; but the original password cannot be obtained from the stored hash

- hash of a message to the original and the recipient can rehash the message and compare it to the supplied hash if they match we can understand message is unchanged this can also be used to confirm no data loss in transmission while data moving and rest also.

- For database administrator digital signatures are rather more involved but in essence, you can sign the hash of a document by encrypting it with your private key, producing a digital signature for the document. Anyone else can then check that you authenticated the text by decrypting the signature with your public key to obtain the original hash again, and comparing it with their hash of the text.

So far that kind of application only used SHA-1 algorithm technique. We have tried as new innovative idea for database at data rest level. Database will get look for different meaning in the aspect of understanding because partial or portion of database will look in digest format rest of them will look in normal format. We can think portion of database looking clearly so we can predict the information very easily but not possible complete identification information will in the format of digest. There won't any limitation for digest count that's depending on developer wish.

Result & Discussion:
We have used java;oracle database and the Java Script coding for generate the SHA-1 algorithm digest message. We have to maintain the separate table for primary key and digest. Digest value appear in the table instead of original values. This illustration attached in the appendix 1 and appendix 2.

Conclusion:
I hope this research will help to the database administrator and big concerns to protect their data's. We have tried our idea in different ways in database that yields good result compare with normal security level. Our method gives better result compare with normal database without security concept.

Future Work:
Every method has its own pros and cons our work also not in exception. In our work slightly memory overhead occurs to avoid this we are trying some new method for resolving this issue. Forthcoming researcher can also try this with different algorithm and some innovative approach in this idea. We also planned to try different methods and different algorithms.

Reference:

[1]R.Agrawal,J.Kiernan,R.Srikant,andY.Xu.AnXPath based preference language for P3P. In Proc. of the 12th Int'l World Wide Web Conference, Budapest, Hungary, May 2003.
[2] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu.Hippocratic databases.In Proc. of the 28th Int'lConference on Very Large Databases, Hong Kong,China, August 2002.
[3]Agrawal,J.Kiernan,R.Srikant,andY.Xu.ImplementingP3P using database technology. In Proc. of the 19thInt'l Con-ference on Data Engineering, Bangalore,India, March 2003.
[4] R. Agrawal and J. Kiernan.Watermarking relational databases.In 28th Int'l Conference on Very Large Databases, Hong Kong, China, August 2002.
[5] G. Hamilton and R. Cattell. JDBC: A Java SQL API.
http://splash.javasoft.com/jdbc/.
[6] A. F. Westin. Freebies and privacy: What net users think. Technical report,Opinion Research Corporation,

http://www.privacyexchange.org/iss/surveys/sr990714.html,
[7] Mattsson, Ulf T.,' A DATABASE ENCRYPTION SOLUTION ,LinuxSecurity.com, 28 July 2004, http://www.linuxsecurity.com/content/view/116068/65/
[8] M. Lindemann and SW Smith, Improving DES Hardware Throughput forShort Operations, IBM Research Report, 2001, http://www.research.ibm.com/secure_systems_department/projects/scop/papers/rc21798.pdf
[9] Oracle Technical White Paper. Database Security in Oracle8i, November 1999.
[10] HOW TO LOCK DOWN ENTERPRISE DATA WITH INFRASTRUCTURE SERVICESOriginally published in (IN)SECURE Magazine – www.insecuremag.comUlf T. Mattsson, CTO Protegrity
[11]Database Encryption - How to Balance Security with Performance Ulf T. MattssonProtegrity Corp.

Appendix :1

Customer ID	Digest
1113	3ca192bd7558780793444f73366c58d60c9d7775
1114	870f1bf229da5eb26e5e5a7c1d69d9451fa7906a
1226	59e221a4758b17362bdeae901c5a8f58057468b2
1334	2d97d80f5af467caf6a638d16c539634e7b1b7a1

Fig 1: Digest maintenance

Appendix:2

Customer ID	Balance	Loan	FD
3ca192bd7558780793444f73366c58d60c9d7775	523892	23456	564534
870f1bf229da5eb26e5e5a7c1d69d9451fa7906a	289311	12341	345342
59e221a4758b17362bdeae901c5a8f58057468b2	121214	50233	231211
2d97d80f5af467caf6a638d16c539634e7b1b7a1	434356	23121	645212

Fig2: Value Appear in Database.