ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

# Tenable Data Access in Untrusted Environment

P.Ramya[1], Dr.Nalini[2], Sundararajan.M[3], Arulselvi S[4]

[1]Lecturer, Dept. of CSE, Bharath University, Chennai, Tamil Nadu, India

[2]Assistant Professor, Dept. of CSE, Bharath University, Chennai, Tamil Nadu, India

[3]Director, Research Center for Computing And Communication, Bharath University, Chennai, India

[4]Co-Director, Research Center for Computing And Communication, Bharath University, Chennai, India

**ABSTRACT***:* The democratization of ubiquitous computing (access data anywhere, anytime, anyhow), the increasing connection of corporate databases to the Internet and the today's natural resort to Webhosting companies strongly emphasize the need for data confidentiality. Database servers arouse user's suspicion because no one can fully trust traditional security mechanisms against more and more frequent and malicious attacks and no one can be fully confident on an invisible DBA administering confidential data. This paper gives an in-depth analysis of existing security solutions and concludes on the intrinsic weakness of the traditional server-based approach to preserve data confidentiality. With this statement in mind, we propose a solution called C-SDA (Chip-Secured Data Access), which enforces data confidentiality and controls personal privileges thanks to a client based security component acting as a mediator between a client and an encrypted database. This component is embedded in a smartcard to prevent any tampering to occur. This cooperation of hardware and software security components constitutes a strong guarantee against attacks threatening personal as well as business data.

## 1. INTRODUCTION

The rapid growth of ubiquitous computing impels mobile users to store personal data on the Web to increase its availability. In the same way, corporate databases are made more and more accessible to authorized employees Over the Internet. Small businesses are prompted to delegate part of their information system to Web-hosting Companies or Database Service Providers (DSP) that guarantee data resiliency, consistency and high availability [eCr02, CaB02, Qck02]. Customer information is also maintained on-line for the needs of e-commerce and ebusiness applications. Typically, Microsoft .NET Passport [Mic02] gathers customer information (identity, passwords, credit card numbers, and profiling data) in an electronic wallet shared by all participating .NET Web sites. Consequently, the amount of sensitive information collected and shared in the marketplace is such that data confidentiality has become one of the major concerns of citizens, companies and public organizations, and constitutes a tremendous challenge for the database community.

Confidential data threatened by attackers is manifold: information related to the private life of individuals (e.g., agenda, address book, bookmarks, medical records, and house hold expenses), credit card numbers, patents, and business strategies, diplomatic or military secrets. Even ordinary data may become sensitive once grouped and well organized in databases. Customers have no other choice than trusting DSP's arguing that their systems are fully secured and their employees are beyond any suspicion. However, according to the "Computer Crime and Security Survey" published by the Computer Security Institute (CSI) and the FBI [FBI01], the theft of intellectual

property due to database vulnerability costs American businesses $103 billion annually and 45% of the attacks are conducted by insiders.

Traditional database security policies rely on user authentication, communication encryption and server enforced access controls [BPS96]. Unfortunately, these mechanisms are inoperative against most insider attacks and particularly against

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

database administrator attacks. Several attempts have been made recently to strengthen Server-based database security policies thanks to database encryption [Ora99, Mat00, HeW01].
This paper first characterizes the intrinsic limits of these server-based solutions with respect to the different types of attacks that can be conducted. With these limitations in mind, we state the dimensions of the *data confidentiality problem*.
While client-based security policies have been historically disregarded considering the vulnerability of client environments [Rus01], we argue that the emergence of smartcard secured client devices fundamentally changes[`1,3]
the problem statement. Initially developed by Bull to secure the French banking system, smartcards have been used successfully around the world in various applications managing secured data (such as banking, pay-TV or GSM subscriber identification, loyalty, healthcare, insurance, etc.). Unfortunately, smartcards suffer from intrinsic hardware constraints that confine their applicability in terms of data management to secure portable folders (e.g., healthcare folder) [ISO99, PBV01].
We capitalize on the security properties of the smartcard to devise a solution to the data confidentiality problem, named *Chip-Secured Data Access (C-SDA)*. C-SDA takes the form of security software embedded in a smartcard. This software acts as an incorruptible mediator between a client and a server hosting an encrypted database. The confidence in C-SDA relies on the fact that data encryption, query evaluation and access right management are insulated in a smartcard and cannot be tampered by anyone, including the cardholder. Dedicated query evaluation techniques are proposed to tackle the strong smartcard hardware constraints. We show the conclusive benefit of associating software and hardware security to preserve data confidentiality.

## II.PROPOSED METHOD

The contribution of this paper is twofold. First, it clearly states the dimensions of the data confidentiality problem and explains to which extent existing security solutions fail in addressing some of these dimensions.[9,12]

*2.1. Data privacy vs. data confidentiality*

This paper concentrates on a particular aspect of database security that is *data confidentiality*. Data confidentiality refers to the ability to share sensitive data among a community of users while respecting the privileges granted by the data owner to each member of the community. Any user external to the community is assumed to have no privilege at all. A special case of data confidentiality is *data privacy*. Data privacy means that the data owned by an individual will never be disclosed to anyone else. Privacy is easier to enforce than confidentiality since sharing is precluded. The simplest and most effective way to ensure data privacy is to encrypt the user's data thanks to a symmetric key algorithm (e.g., DES [NIS93]). The user being the unique holder of the cipher key, no one else can access the clear text form of the data. Several Storage Service Providers propose to manage encrypted backups for personal data [Sky02]. They guarantee that data is encrypted at all times from transmission of a customer's computer to their server and back and remains safe from unauthorized access even by their staff.
Data privacy solutions cover only a restricted range of applications considering that even private data is subject to sharing (e.g., patient's medical records are shared by doctors, customer's information is shared by e-commerce sites). Thus, the remainder of the paper focuses on the more general problem of data confidentiality and places much emphasis on access right management.

*2.2. The attackers*

In the light of the preceding section, we can identify three classes of attackers that can threaten data confidentiality:
- *Intruder:* a person who infiltrates a computer system and tries to extract valuable information from the database footprint on disk (DBMS access controls are bypassed).[4,8]
- *Insider*: a person who belongs to a community of users properly identified by the computer system and the database server and who tries to get information exceeding her own access rights.
- *Administrator*: a person who has enough (usually all) privileges to administer a computer system (System Administrator) or a DBMS (Database Administrator or DBA). These privileges give her the opportunity to

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

tamper the access right definition and to spy on the DBMS behavior. An Intruder who usurps successfully the identity of an Insider or an Administrator will be considered as such in the rest of the paper.

*2.3. Weaknesses of server-based security policies*

Traditional database security policies rely on three well established principles [BPS96]: (1) user identification and authentication, that can be supported by mechanisms ranging from simple login/password methods up to smartcard or biometrics device-based methods;

(2) network encryption, that guarantees the confidentiality and the integrity of client/server communications; and (3) server-enforced access control and privilege management. Although these mechanisms are clearly required, they fail to answer all threats identified earlier for two obvious reasons. The first reason is that the confidence on the server never exceeds the confidence the user is ready to place in the DBA. This confidence may vary depending on the users, the Web-hosting companies or the countries but, as far as data confidentiality is concerned, this confidence is generally quite low. The second reason is the increasing number of commercial or institutional sites that are hacked, demonstrating the difficulty of making the hosting computing system secure enough to prevent any intrusion.[13,15]

Recent attempts have been made to reinforce the server security by encrypting the database. Some commercial DBMSs provide encryption packages to this end [Ora00]. However, if encryption provides an effective answer to attacks conducted on the database footprint by an Intruder, it does not enforce data confidentiality on its own. Indeed, the server being still responsible for query execution and access right management, encryption makes just a bit more tedious the Administrator attacks. In these solutions, the management of cryptographic keys is under the application's responsibility and data is decrypted on the fly by the server at query evaluation time. Thanks to her privileges and to the DBMS auditing tools, the DBA can change the encryption package, can get the cryptographic keys, can modify the access right definition and can even snoop the memory to get the data while it is decrypted.

The proliferation of solutions to increase database security exemplifies the acuity of the problem. However, existing solutions fail in answering the data confidentiality requirements listed below:[16,19]

**Data Confidentiality Requirements**

1. confidential data must be managed by an auto administered DBMS to cast off the DBA privileges,
2. this DBMS must be hosted by an auto-administered computing system to cast off the system administrator privileges,
3. this computing system must constitute a Secure Operating Environment (SOE)1 to cast off any Intruder action.

The traditional database server approach suffer from a strong handicap to meet these requirements because existing DBMSs, as well as the computing systems they rely on, are far too complex, first to be auto-administered and second to constitute a SOE. The first assumption is strengthen by the analysis done in [ChW00] which measures the distance separating current technologies from future self-tuning and zero-admin DBMSs2. The worrying numbers regularly published by the Computer Security Institute and the FBI on database vulnerability [FBI01] truly confirms the second assumption.

*2.4. Client-based security policies*

The weaknesses of the server-based approach to meet the data confidentiality requirements led us to devise client based solutions. As a preliminary remark, let us notice that the solution presented in section 2.1 to enforce data privacy is typically client-based since the server does nothing but storing encrypted data. Unfortunately, these solutions do not support sharing. Enforcing data confidentiality in a client-based approach means delegating the sharing control to the client devices. However, client-based approaches have been historically disregarded considering that users have themselves the opportunity to hack the client system, and then the sharing control in our context, with total impunity [Rus01].

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

The emergence of smartcard secured client equipment's drastically changes these conclusions. We illustrate the *smartcard-client-based* approach below through practical examples, and discuss to which extent they meet the data confidentiality requirements.

Smartcard is undoubtedly the most secure and cheap computing device today. The strength of smartcard applications regarding data confidentiality is threefold: (1) existing smartcard applications are simple enough to require zero-administration once downloaded in the card, (2) thanks to its hardware architecture making tampering extremely difficult [AnK96, ScS99], the smartcard is probably the best representative of SOE, (3) the high cost of an attack and its practical difficulty (holding the card) must be weighted up with its benefit (the data of a single user can be revealed). A common assumption is that a system can be considered secure if the cost of hacking it exceeds the value of the disclosed information. Conversely, the cost of security for the user is negligible considering the price of a smartcard (a few dollars).

Interesting attempts have been made to push away the smartcard storage limit. The first solution, due to the *Web Card* project [Van98], consists of storing in the smartcard URLs referencing huge, but unprotected, external data. *The Vault* [Big98] extends the Web Card approach by encrypting the documents referenced by URLs. Undoubtedly, the Vault meets the requirements of some applications but it does not constitute a solution from the database point of view. Indeed, the on-board database is seen as a catalog of large encrypted documents rather than as a regular database holding numerous fine-grain objects that can be shared and queried.[20,25]

*2.5. Problem definition*

From the preceding discussions, we can identify the different dimensions of the *data confidentiality problem* addressed in this paper.

*Data confidentiality problem*

- *Privacy and confidentiality*: privacy of personal data and confidentiality of shared data must be guaranteed against attacks conducted by Intruders, Insiders and Administrators.
- *Storage capacity*: the system must not limit the volume nor the cardinality of the database.
- *Sharing capacity*: if required, any data may be shared among multiple authorized users.
- *Query capacity*: any data, whatever its granularity, may be queried through a predicate-based language (typically SQL).[26,28]
- *Pertinence*: the system must guarantee an acceptable response time to each user, must be scalable and must be economically viable to meet the requirements of large public applications.

## III. C-SDA BASELINE

Before discussing the principles of Chip-Secured Data Access (C-SDA), we first analyze how smartcard client based solutions answer each dimension of the data confidentiality problem:

- *Privacy and confidentiality:* enforced by the fact that the smartcard is a SOE hosting the data as well as the DBMS engine and that this DBMS is self or user administered.
- *Storage capacity*: limited by the smartcard stable storage capacity.
- *Sharing capacity*: limited by the need to share physically the same card4.
- *Query capacity:* depends on the power of the embedded database engine. While query capacity is limited to simple selection in the SCQL standard [ISO99], PicoDBMS [PBV01] demonstrates the feasibility of embedding powerful query engines supporting selection, join, grouping and aggregate calculus.
- *Pertinence:* well suited in terms of performance (the smartcard DBMS is mono-user and works on a reduced set of data), of scalability (one smartcard per user) and of price (a few dollar per smartcard).

Given these statements, solving the data confidentiality problem sums up to bypass the storage and sharing limitations without hurting the other dimensions. The concept of server typically addresses the storage and sharing issues. Thus, let us consider to which extent the sphere of security provided by the smartcard could be extended to a remote server holding encrypted data. As discussed in section 2.3, the first security breach of the server-based approach comes from the fact that

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

data is decrypted by the server at query execution time. Assuming that the DBMS query engine remains hosted by the smartcard, this eliminates the need to decrypt data on the server side. The second security breach of the server-based approach comes from the fact that access rights are enforced by the server and administered by an untrusted DBA. Let us assume that the DBMS access right manager remains hosted by the smartcard, the DBA (or an Intruder usurping her identity) is no longer able to abuse them.

Can we infer from the preceding assumptions that a server acting as an encrypted repository for a smartcard.

In the same spirit, since the data that flows from the server to the smartcard DBMS is encrypted, can we infer that the communication channel is part of the smartcard's sphere of security? Again, the answer is 'no' since the

Communication channel may undergo several forms of attacks. At first sight, privacy and confidentiality are preserved anyway. However, an Insider may compare the encrypted data issued from the server with the query result

that appears in plain text on its terminal. This may help her to conduct a *known plain text cryptanalysis* in order to

deduce the encryption keys hosted by the smartcard. Thanks to these keys, the Insider may attempt to access data exceeding her own access rights. Indeed, the Insider may have the privilege to see the result of a query computed by the smartcard DBMS on data that is outside the scope of her privilege6. Consequently, re-encrypting the communication with a session key protocol (e.g., SSL) is necessary to enforce confidentiality7.

The baseline of C-SDA is then to build a sphere of confidentiality encompassing the smartcard DBMS, the server and the communication channel linking them. The resulting functional architecture is pictured in Figure 3 and roughly works as follows. Each smartcard is equipped with a database engine managing access rights, query evaluation and encryption. When the user issues a query, the smartcard DBMS first checks the user's access rights

and, in the positive case, gets the data from the server, decrypts it, executes the query and delivers the result to the terminal.[29,30].

## IV. QUERY MANAGEMENT

In order to evaluate the technical soundness of the C-SDA architecture in terms of query evaluation feasibility and efficiency, we first recall the smartcard characteristics that are relevant to this issue. Then, we propose a query evaluation principle that matches these smartcard characteristics whatever the volume of data involved in a query.

### *4.1. Smartcard characteristics*

Current smartcards include in a monolithic chip, a 32 bits RISC processor at about 30 MIPS, memory modules (of about 96 KB of ROM, 4 KB of static RAM and 128 KB of EEPROM), a serial I/O channel (current bandwidth is around 9.6Kbps but the ISO standard allows up to 100Kbps) and security components preventing tampering [ISO98]. ROM is used to store the operating system, fixed data and standard routines. RAM is used to manage the execution stack of programs and to calculate results. EEPROM is used to store persistent information. EEPROM has very fast read time (60-100 ns/word) comparable to RAM, but a dramatically slow write time (1to 5 ms/word).

The main constraints of current smartcards are therefore: (i) the very limited storage capacity; (ii) the very slow write time in EEPROM and (iii) the extremely reduced size of the RAM. On the other hand, smartcards benefit from a very high security level and from a very powerful CPU with respect to the other resources. This makes the smartcard an asymmetric computing architecture which strongly differs from any other computing devices.[31,32]

### *4.2. Query evaluation principle*

A naive interpretation of the C-SDA architecture depicted in Figure 3 is to consider that the server acts as a persistent encrypted virtual memory which is accessed by the smartcard DBMS during query evaluation, any time a data item is requested for computation. Such an architecture would suffer from disastrous performance because it would incur a prohibitive communication cost (one call per data item) and I/O cost (traditional server optimizations

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

Become irrelevant). It may even happen that the same data be loaded several times from the server if the smartcard DBMS cannot keep enough local resources to cache it. Last but not least, the smartcard hardware constraints impose to design very specific query evaluation strategies. While ad-hoc strategies have been shown convenient in the context of small-embedded databases, their algorithm complexity renders them totally inappropriate for large databases [PBV01].
Thus, new query evaluation strategies that better exploit the computational resources available on the server and even on the terminal must be devised. This leads to split a query Q into a composition of the form Qs ° Qc ° Qt, where Qs, Qc and Qt denote respectively the sub-query evaluated on the server, the card and the terminal. The imbalance between the smartcard, the server and the terminal in terms of computing resources advocates pushing the biggest part of the computation down into Qs and Qt. However, the imbalance between these same components in terms of security leads to the following compromise:

□ □*Server subquery (Qs):* the server must execute the largest part of the query as far as confidentiality is not compromised. That is, any predicate that can be evaluated on the encrypted form of the data must be pushed down to the server. To simplify things, we consider below that predicates based on an equality comparator {=, ≠} satisfy this condition9. In the sequel, we call these predicates *equi-predicates* in opposition to *inequi-predicates* based on inequality operators {>, ≥, <,≤}.
□ □*Smartcard subquery (Qc):* the smartcard DBMS is responsible for filtering the result of Qs to evaluate all predicates that cannot be pushed down to Qs and for computing aggregation functions if required. The terminal cannot participate to this evaluation because the data flow resulting from Qs may go beyond the user's access rights.
□ □*Terminal subquery (Qt):* due to the confidentiality consideration mentioned earlier, the terminal can only evaluate the part of the query related to the result presentation. Typically, it can handle the sort and the distinct operators, if requested by the user.
The challenge in decomposing Q into Qs ° Qc ° Qt is twofold. First, the global evaluation must meet the *pertinence* requirement in terms of performance and scalability. Second, Qc must accommodate the smartcard's hardware constraints. Query evaluation on the smartcard precludes the generation of any intermediate results since: (i) the RAM capacity cannot accommodate them, (ii) RAM cannot overflow into EEPROM due to the dramatic cost of
EEPROM writes and (iii) intermediate results cannot be externalized to the terminal without hurting confidentiality.

## V. CONFIDENTIALITY AND ENCRYPTION

This section fixes a set of encryption rules required to answer accurately the data confidentiality problem. Then, it shows how the smartcard device can be exploited to increase the privacy and confidentiality of a reduced set of highly sensitive data. Finally, it addresses the management of access rights and concludes with a discussion on the limits of the solution.

### *5.1. Database encryption*

From the beginning of the paper, we have considered implicitly that the whole database was encrypted. Obviously, only the confidential part of it needs to be encrypted. For the sake of simplicity, we will not discuss further the cohabitation between clear and encrypted data because it does not present a major technical difficulty. Thus, we concentrate in the sequel on the quality of the database encryption.
As stated in section 3, the level of confidence placed in C-SDA is strongly related to the confidence placed in the data encryption strategy. In our context, the following *data encryption rules* apply:

***Key insulation rule:*** *encryption keys must remain confined in the smartcard.*

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

This rule is required to prevent any attack conducted by the DBA, an Intruder and even an Insider. Consequently, data encryption and decryption must be handled by the smartcard as well. Note that the cardholder herself has no way to access the encryption keys hosted by its own card. These keys remain under the exclusive control of the in card C-SDA software.
***Sharing rule:*** *encryption must remain orthogonal to access rights.*

As explained in section 2.1, encryption alone is sufficient to implement data privacy, assuming that each user encrypts her own data with a secrete key. Thus, encryption acts as a binary access right granting or revoking all privileges to the user depending on whether or not she knows the secrete key. On the contrary, data confidentiality requires sharing the same key(s) among a community of authorized users. Unfortunately, there is no bijection between encryption and access rights because these two mechanisms do not operate at the same level of granularity. Access rights are commonly attached to database views to share data at a very fine-grain level. The sharing is thus predicate-based. Achieving the same level of sharing with encryption alone would require defining as many encryption keys as possible SQL qualifications. Access rights can even be defined on virtual data (e.g., aggregate calculus) that obviously cannot be encrypted. Consequently, encryption rules must remain orthogonal to access right management. Assuming key Ki is used to encrypt data shared among multiple users, Ki must be hosted by the smartcard of each of these users but the key usage is restricted to the in-card C-SDA software that controls access rights.

***Computation rule:*** *encryption must preserve attribute equality comparisons.*
Encrypting the database on a tuple, column, or relation basis precludes any computation to occur on the server side without decrypting the data first. Thus, the encryption must be done on an attribute basis. In addition, as stated in section 4.2, the minimal assumption required to allow server computation without decryption is $\forall di,dj,\ E(di) = E(dj) \Leftrightarrow di = dj$. Obviously, this assumption is required only for couple of data that may be subject to comparison.

***Performance rule:*** *encryption must be symmetric and client-based.*
As stated in section 4.2, client-based encryption/decryption is the first guarantee of scalability. Moreover, considering the large volume of data to be encrypted/decrypted, we promote the use of symmetric encryption algorithms (e.g., DES) because they are more robust and much more efficient (three orders of magnitude faster) than Asymmetric algorithms (e.g., RSA [RSA93]). The secure diffusion of secrete keys is the major problem of symmetric algorithms in traditional architectures. This problem is solved by nature in the C-SDA context, thanks to the smartcard device that provides a secure key hosting. Thus, keys are distributed among users along with smartcards.

***Multi-key encryption rule:*** *encryption must exploit as much different keys as possible.*
Increasing the number of keys in the encryption process has two main advantages. First, it makes statistical attacks more difficult to conduct. Second, it reduces the amount of data that will be disclosed if the aforementioned attack succeeds. Different techniques can be envisioned to use multiple keys while respecting the computation rule. A first solution is based on vertical fragmentation that is encrypting with different keys the columns that will never participate to an equi-join (e.g., Person.name and Person.age). A second solution is based on horizontal partitioning, that is encrypting with different keys the attribute values of the same column thanks to a one-way hash function. For instance, *Key(h(a))* can be used as a parameter to encrypt the attribute value *a*, and the value *(h(a), Ekey(h(a))(a))* is stored in the database in place of *a*. Note that this solution respects the computation rule. Other techniques may be used but space limitations forbid their presentation in this paper.

*5.2. Sensitive data*

The persistent storage capacity of the smartcard introduces new alternatives to achieve data privacy and confidentiality. Basically, highly sensitive data may be stored in the smartcard instead of in the server, thereby making it ultimately robust against attacks. For instance, identification information could benefit from this property (e.g., name,

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

social security number …), so that the database in the server is depersonalized. This technique however introduces three issues: (i) how to integrate this sensitive data in the query evaluation process, (ii) how to guarantee its durability and (iii) how to share it if it is used by multiple users.
To make the integration of sensitive data in the query evaluation process as simple as possible, we propose to group sensitive data in *sensitive domains* (i.e., set of data items without duplicates) and to store indices referencing these domain values in place of the corresponding data in the server. This technique can be formally considered as a particular encryption method *E(data)→domain index* that is definitely unbreakable without the smartcard.

*5.3. Access Right Management*
As stated in section 3, access right management must be embedded in the smartcard to prevent any DBA tampering.
Since access rights are commonly defined on database views, the views have to be managed by the smartcard as well. This raises the problem of access rights and views evolution. If the smartcard is responsible for controlling access rights and views, their definitions have to be securely stored in a server accessible by all smartcards. Modeling the list of access right definitions and the list of database view definitions as two dynamic and shared sensitive domains brings a simple and accurate solution to this problem. The crucial question regarding access rights is who is responsible for granting/revoking them. The common rule in database systems is that the owner of an object inherits this responsibility. In practice, the unlimited privileges of the DBA contradict this rule. Using C-SDA, the DBA conserves all her privileges, so that she can administer the database server but she has no way to break the data confidentiality, as long as she has no access to the user's smartcard. As a conclusion, a C-SDA user is definitely the unique holder of her data and she decides if she wants to exhibit them and to whom.

5.4. Limits of the solution
One may wonder whether this combination of hardware (the smartcard) and software (C-SDA) security components constitutes the ultimate protection against data confidentiality attacks. In this respect, we must state the limits of the solution.
First, an Intruder can infiltrate the user's terminal in order to snoop the query results that are presented in plain text to the user or to alter the query expression sent by the terminal to the smartcard before processing. By this way, the Intruder may try to execute a query selecting more data than expected by the user and snoop them. Anyway, such attack can reveal only data being in the user's access right scope. This threat cannot be avoided by any security
Architecture, unless the terminal is itself secure. To secure the terminal, both the screen and the keyboard must be part of the SOE, like in today's smartcard payment devices. This solution can be suitable for users willing only to consult their data but is inadequate as soon as computation is required on these data.

## VI. C-SDA SCENARIO

This section presents a complete C-SDA scenario illustrating the step by step evaluation of a simple query on a corporate database. Confidentiality and performance issues are discussed along the scenario unfolding.

*6.1. Query Execution with C-SDA*
Let us consider a business database application where the invoice department is willing to bill invoices having a total amount greater than 1000 US$. The privilege of the invoice department clerk is assumed to be restricted to the *select* operation on the view *Invoice*. This view calculates for each customer, the total amount of delivered orders since January 2002. This view prevents an untrusted clerk to access confidential order-lines. Figure 6 shows a query *Q*, issued by the clerk and expressed on the view Invoice, and the query *Q'*, resulting from the view resolution and expressed on the base relations Customer and Order. The execution of query *Q* comprises the following steps.
1. *Metadata refreshing:* At connection time (i.e., when the user inserts her smartcard to the card reader), C-SDA

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

contacts the durability server(s) in order to refresh its local copy of relation and view definitions, access right information's and sensitive data.

2. *Access Right checking and view resolution:* The *access right manager* checks that query *Q* involves only authorized relations and views. Then, the *view manager* merges *Q* with the view definition to produce *Q'*.

3. *Query splitting:* The *query splitter* splits query *Q'* into *Qs* (step 4), *Qc* (step 6) and *Qt* (step 7) conforming to the decomposition principle detailed in section 4.2. *Qs* is then rewritten in an "encrypted SQL form", that is relation names, attributes and constant are encrypted (encryption is denoted by *E()* in Figure 6). Note that the encrypted form of a well-formed SQL query is a well formed SQL query.

4. *Qs transmission and execution:* The encrypted query *Qs* is sent to the database server using a secured communication protocol. Secured communication is mandatory to avoid any falsification of *Qs* before transmission (which may permit a malicious user to access more data than granted). The database server optimizes and processes *Qs* as any traditional query, without being aware of encryption. The query execution plan of *Qs* is pictured in Figure 7.

5. *Qs Result transmission:* The encrypted result *Rs* is sent back to the smartcard using a secured communication protocol. As explained in section 3, secured communication is mandatory here to avoid plaintext cryptanalysis on the terminal.

.

*6.2. Optimization Issues*

The performance problem pointed out in section 4.2 is exemplified in this scenario. Assume that only 1% of orders satisfies the selection on *date*, 99% of the *Rs* tuples sent back to the smartcard are irrelevant, generating a bottleneck on the smartcard input-channel. In the following we sketch a solution alleviating this problem. Other optimizations of the C-SDA architecture can undoubtedly be devised but are out of the scope of this paper. Finally, the smartcard query splitter adds the semi-join predicate T.E(date) = Order.E(date) to the initial query *Qs* and sends it to the server for computation (see Figure 8). This strategy applies as well to inequi-join predicates, and can be exploited iteratively for all inequi-predicates involved in the same query.

## VII. CONCLUSION AND FUTURE PROSPECTS

The tremendous development of Internet applications prompts citizens and companies to put always more data accessible through the Web. Preserving data confidentiality in this context is becoming one of the most challenging issues for the database community. This paper addresses this issue and makes the following contributions. First, it gives an in-depth analysis of the security solutions proposed in the database field and capitalizes on strengths and weaknesses of these approaches to clearly state the dimensions of the *data confidentiality problem*. Second, it proposes the *Chip-Secured Data Access* (C-SDA) principle as a solution to this problem. The main idea underlying C-SDA is to insulate data encryption, query evaluation and access right management in a Secured Operating Environment (SOE). Third, query evaluation and optimization techniques are proposed to tackle the strong hardware constraints introduced by the most popular representative of SOE, the smartcard's-SDA is being validated in the context of a B2B project founded by the French ANVAR agency. This project, started in january 2002, aims at sharing an EDI database between business partners. Depending on the business model, this database can be hosted by a DSP or by one of the partner, but the data confidentiality requirements remain the same. C-SDA has been devised in the context of smartcards because of its wide acceptance and its well-established technology. However, the C-SDA architecture can be adapted to other secured computing devices. For instance, the Dallas i-button [iBu02] provides a security level comparable to smartcards but benefits from a higher bandwidth with the terminal. Such technology could be exploited to alleviate the performance problem induced by inequi-predicates. The apparition of high-end secure coprocessor [Swe99] may, in the future, render viable tamper-resistant server-based solutions that are technically unfeasible today for performance and scalability reasons. In all situations, the interactions between the C-SDA software hosted by the secured device and the encrypted data store will remain the same, but with different technical tradeoffs.Other important open issues concern the extension of C-SDA to

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

more complex data models, query languages and client/server interactions. More generally, we believe that tamper-resistant devices will have an increasing influence on the way security solutions for information systems will be devised.

## REFERENCES

[1] N. Anciaux, C. Bobineau, L. Bouganim, P. Pucheral, P.Valduriez, "PicoDBMS: Validation and Experience", *Int. Conf. on VLDB*, 2001.
[2] R. Anderson, M. Kuhn, "Tamper Resistance – aCautionary Note", *USENIX Workshop on Electronic Commerce*, 1996.
[3] P. Biget "The Vault, an Architecture for Smartcards to Gain Infinite Memory", Smart Card Research and Advanced Application Conference (CARDIS'98), 1998.
[4] M. Blaze, "High-Bandwidth Encryption with Low-Bandwidth Smartcards", AT&T Bell Labs, 1995. (ftp://ftp.research.att.com/dist/mab/ card cipher.ps)
[5] A. Baraani, J. Pieprzyk, R. Safavi-Naini "Security In Databases: A Survey Study", 1996. citeseer.nj.nec.com/baraani-dastjerdi96security.html
[6] The Caspio Bridge DSP. www.caspio.com/bridge.htm[ChW00] S. Chaudhuri, G. Weikum, "Rethinking Database System Architecture: Towards a Self-tuning RISC-style Database System", Int. Conf. on VLDB, 2000.
[7] J. Domingo-Ferrer, "Multi-application smart cards and encrypted data processing", Future GenerationComputer Systems, (13), 1997.
[8] The eCriteria DSP. www.ecriteria.net
[9] Computer Security Institute, "CSI/FBI Computer Crime and Security Survey". www.gocsi.com/forms/fbi/pdf.html
[10] J. He, M. Wang, "Cryptography and Relational Database Management Systems", Int. Database and Engineering and Application Symposium, 2001.
[11] The crypto iButton with Java - (http://www.ibutton.com/)
[12] International Standardization Organization (ISO), *Integrated Circuit(s) Cards with Contacts – Part 1: Physical Characteristics*, ISO/IEC 7816-1, 1998.
[13] International Standardization Organization (ISO), *Integrated Circuit(s) Cards with Contacts – Part 7:Interindustry Commands for Structured Card Query Language (SCQL)*, ISO/IEC 7816-7, 1999.
[14] U. Mattsson, *Secure.Data Functional Overview*, Protegity Technical Paper TWP-0011, 2000. (http://www.protegrity.com/White_Papers.html)
[15] The Microsoft.Net Passport. www.passport.com
[16] National Institute of Standards and Technology, *Announcing the Data Encryption Standard (DES)*, FIPSPUB 46-2, 1993.
[17] National Institute of Standards and Technology, Announcement of Weakness in the Secure Hash Standard, 1994.
[18] Oracle Corp., *Database Security in Oracle8i*, 1999. otn.oracle.com/deploy/security/oracle8i
[19] Oracle Corp., *Advanced Security Administrator Guide*, Release 8.1.7, 2000.
[20] P. Pucheral, L. Bouganim, P. Valduriez, C. Bobineau, "PicoDBMS: Scaling down Database Techniques for the Smartcard", *VLDB Journal*, 10(2-3), 2001.
[21] The Quickbase DSP. https://www.quickbase.com/
[RAD78] R. L. Rivest, L. Adleman and M. L. Dertouzos, "On Data Banks and Privacy Homomorphisms", Foundations of Secure Computation. Academic Press, 1978.
[22] RSA Laboratories, *PKCS #1: RSA Encryption Standard*, RSA Laboratories Technical Note, 1993.
[23] Ryan Russel et al., *Hack Proofing Your Network*, Syngress Publishing, 2001.
[24] B. Schneier, A. Shostack, "Breaking up is hard to do: Modeling Security Threats for Smart Cards", *USENIX Symposium on Smart Cards*, 1999.
[25] Anbazhagan R., Satheesh B., Gopalakrishnan K., 'Mathematical modeling and simulation of modern cars in the role of stability analysis", Indian Journal of Science and Technology, ISSN : 0974-6846, 6(S5) (2013) pp.4633-4641.
[26] Muruganantham S., Srivastha P.K., Khanaa, "Object based middleware for grid computing", Journal of Computer Science, ISSN : 1552-6607, 6(3) (2010) pp.336-340.
[27] B. Schneier, *Applied Cryptography*, 2nd Edition, John Wiley & Sons, 1996.
[28] Sengottuvel P., Satishkumar S., Dinakaran D., "Optimization of multiple characteristics of EDM parameters based on desirability approach and fuzzy modeling", Procedia Engineering, ISSN : 0975 – 7384, 64() (2013) pp.1069-1078.
[29] SkyDesk : @Backup. www.backup.com/index.htm
[30] S.W. Smith, S.H. Weingart, Building a High- Performance, Programmable, Secure Coprocessor, Computer Networks (31) – 1999
[31] Langeswaran K., Revathy R., Kumar S.G., Vijayaprakash S., Balasubramanian M.P., "Kaempferol ameliorates aflatoxin B1 (AFB1) induced hepatocellular carcinoma through modifying metabolizing enzymes, membrane bound ATPases and mitochondrial TCA cycle enzymes", Asian Pacific Journal of Tropical Biomedicine, ISSN : 2221-1691, 2(S3)(2012) pp.S1653-S1659.
[32] Rajendran S., Muthupalani R.S., Ramanathan A., "Lack of RING finger domain (RFD) mutations of the c-Cbl gene in oral squamous cell carcinomas in Chennai, India", Asian Pacific Journal of Cancer Prevention, ISSN : 1513-7368, 14(2) (2013) pp.1073-1075.
[33] Bharthvajan R, Human Resource - Strategy and Outsource, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 15273-15276, Vol. 3, Issue 8, August 2014
[34] Bharthvajan R, Human Resource Management and Supply Chain Management Intersection, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 10163-10167, Vol. 3, Issue 3, March 2014

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 4, April 2015**

[35] Bharthvajan R, Women Entrepreneurs & Problems Of Women Entrepreneurs, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 16105-16110, Vol. 3, Issue 9, September 2014
[36] Bharthvajan R, Organizational Culture and Climate, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 8870-8874, Vol. 3, Issue 1, January 2014
[37] C.Rathika Thaya Kumari , Dr.A.Mukunthan, M.Nageshwari, Electric and Magnetic Properties of Semiconductors and Metals in One, Two and Three Dimensions, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 271-279, Vol. 2, Issue 1, January 2013
[38] C.Tamil Selvi & Dr. A. Mukunthan, Different Varieties of Plantain (Banana) and Their Estimation by Chemical Tests, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753,pp 1099-1105, Vol. 2, Issue 4, April 2013