Closing the Key Loophole in MLS Databases

Nenad Jukic
Loyola University Chicago
njukic@luc.edu

Svetlozar Nestorov
The University of Chicago
evtimov@cs.uchicago.edu

Susan Vrbsky
The University of Alabama
vrbsky@cs.ua.edu

Abstract: There has been an abundance of research within the last couple of decades in the area of multilevel secure (MLS) databases. Recent work in this field deals with the processing of multilevel transactions, expanding the logic of MLS query languages, and utilizing MLS principles within the realm of E-Business. However, there is a basic flaw within the MLS logic, which obstructs the handling of clearance-invariant aggregate queries and physical-entity related queries where some of the information in the database may be gleaned from the outside world. This flaw stands in the way of a more pervasive adoption of MLS models by the developers of practical applications. This paper clearly identifies the cause of this impediment – the cover story dependence on the value of a user-defined key – and proposes a practical solution.

1. Introduction

Since the advent of modern database management systems, the need to ensure the security and proper access control for databases that contain sensitive information has never been greater. The multilevel secure (MLS) relational data models [3] [5] [8] [11] [14] [16] can provide a wholesale solution for this need in environments that exhibit hierarchical propagation of information (such as military institutions, government agencies, airlines or hospitals).

There has been an abundance of research within the last couple of decades in the area of MLS databases. Recent work in this field deals with the processing of multilevel transactions [1] [17], expanding the logic of MLS query languages [6], and utilizing MLS data models within the realm of E-Business [9]. Even though a great deal of contemporary MLS research focuses on the implementation and application issues, there is still an overlooked fundamental shortcoming within the basic MLS principles which stands in the way of a more pervasive adoption of the model by the developers of practical applications. We identify this shortcoming as *the cover story dependence on the value of a user-defined key,* which we shall call *the key loophole*.

In MLS relational databases multiple records on various security levels can depict the same real-world entity. For such records non-key attributes can have different values at different security levels. Providing information to users at lower security levels that is different from the information stored at higher security levels is called a *cover story*. Cover stories provide a mechanism to protect information that should only be known to users at higher security levels from users at lower levels. Until now, in every MLS database key attributes are required to have the same value at all security levels. This requirement excludes the possibility of users at different security levels from seeing different values for the key attributes, even though there are applications for which it is necessary to provide a cover story for the key attributes (in order to mask the value of an identifier of a depicted object to users at lower security levels.)

In this paper we will analyze this drawback, identify its implications, and propose a solution.

2. MLS Data Models

MLS models are based on the classification of the system elements, where classifications are expressed by security levels. Data objects have security levels and users have clearance levels. The security levels of objects are also known as security labels. A security label can contain one security level or a list of levels. As an example, we can have three possible classifications S-Secret, C-Classified, and U-Unclassified, where S is a higher classification than C and U, and C is a higher classification than U. A security (or clearance) level l1 *dominates* another level l2 (stated as l1 ≥ l2), if l1 is higher than or on the same level as l2 in the partial (or total) order of security levels. For example, S ≥ C ≥ U. According to the Bell-LaPadula [2] simple property, a subject (user) can read a certain object (data) only if the subject's clearance level dominates the object's security level. In other words, a subject cannot read an object at a higher or incomparable security level than the subject. A second restriction on multilevel secure databases [2] is the *-property, which states that all writes take place at the subject's security level or higher.

Many MLS relational database models have been proposed and early work in MLS relational databases focused on the semantics and the relational algebra for such models. The SeaView model [3] was the first formal MLS secure relational database designed to provide mandatory security protection. The Sea View model extended the concept of a database relation to include the security labels. A relation that is extended with security classifications is called a multilevel relation. The Jajodia-Sandhu model [8] was derived from the SeaView model. It was shown in [7] that the SeaView model can result in the proliferation of tuples on updates and the Jajodia-Sandhu model addresses this shortcoming. The Smith-Winslett model [16] was the first model to extensively address the semantics of an MLS database. The MLR model [14] is substantially based on the Sandhu-Jajodia model, and also integrates the belief-based semantics of the Smith-Winslett model.

In our previous work, we have shown that all of the aforementioned models can present users with some information that is difficult to interpret [10]. Consequently, we have developed the BCMLS model [11], which addressed those concerns by including the semantics for an unambiguous interpretation of all data presented to the users.

To illustrate the MLS models we will use the following example scenario. The notation used is taken from the BCMLS model.

Example 1.
Midtown Hospital keeps track of its resident patients in an MLS database (Figure 1). The hospital classifies its employees into three clearance categories: U, C, and S, which determines the sensitivity of information they are allowed to see. Every residential patient must be accounted for, on every clearance level. However the correct patient's age and diagnosis may have to be hidden from some security levels. The sample table shown in Figure 1 contains information about two patients. All the information about patient Alan Jones is available for all three clearance levels. However, the information about patient Diva Megastar is more sensitive. The subjects on the S level see correctly her diagnosis, age, and room number. The subjects on the C level see her correct diagnosis and room number, but her age is masked by a cover story for age. The subjects on the U level see her correct room number, but they are given a cover story for both her diagnosis and her age.

MIDTOWN HOSPITAL PATIENTS TABLE

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
Diva Megastar UCS	Exhaustion U-CS	32 UC-S	201 UCS	U-CS
Diva Megastar UCS	Intoxication CS	32 UC-S	201 UCS	C-S
Diva Megastar UCS	Intoxication CS	42 S	201 UCS	S

Figure 1

Within a BCMLS table each attribute is accompanied by the security label that can contain more than one letter. The first letter in the label indicates the security level on which the value of the attribute was created. Such a level is called the primary level of that attribute. Labeled information is always believed to be true by the users whose clearance is equivalent to the security level indicated by the primary level of the label. The letters that follow the first letter of the label indicate the security levels where users from those levels do have a belief about labeled information, but the labeled information was not created at that level. Such levels are called secondary levels. Each security level in the label must dominate the level to its left. Letters that are not preceded by the "-" symbol indicate the secondary levels where the information is believed to be true. The letters following the "-" symbol indicate the secondary levels where the information is believed to be false. In addition to labeling each attribute with a security label, a tuple as a whole is also labeled by a security label, which is depicted by the TC column. The tuple is visible on a certain level only if the TC label contains the label for that level. Also, not every part of the label is visible to every user. Only the parts of the label that depict the user's levels or levels below are visible. Figure 2 shows how the table shown in Figure 1 would be seen by the users from three different levels.

MIDTOWN HOSPITAL PATIENTS TABLE – U View

Patient Name	Diagnosis	Age	RoomNo
Alan Jones	Exhaustion	56	101
Diva Megastar	Exhaustion	32	201

MIDTOWN HOSPITAL PATIENTS TABLE – C View

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UC	Exhaustion UC	56 UC	101 UC	UC
Diva Megastar UC	Exhaustion U-C	32 UC	201 UC	U-C
Diva Megastar UC	Intoxication C	32 C	201 UC	C

MIDTOWN HOSPITAL PATIENTS TABLE – S View

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
Diva Megastar UCS	Exhaustion U-CS	32 UC-S	201 UCS	U-CS
Diva Megastar UCS	Intoxication CS	32 UC-S	201 UCS	C-S
Diva Megastar UCS	Intoxication CS	42 S	201 UCS	S

Figure 2

In MLS relations multiple tuples can exist at different security levels representing contradictory information about the same entity. Assume a user is at security level c. If a lower level tuple with a TC< c, represents the same entity as some other higher level tuple, where TC = c, the lower level tuple is interpreted by a higher level user as a false tuple that represents a cover story [4] [15] for the entity represented by the higher level tuple. A belief held by c-level users that a lower level tuple is a cover story tuple is derived from the fact that there exists a c-level tuple that represents the same entity as the cover story tuple. Every user on the higher-level c has the following belief about the cover story lower-level tuple: *"Some attribute values of this lower level tuple incorrectly represent a real-world entity."* In Figure 2 (S-view), S level users see the second and third tuples as a cover story of the fourth tuple. Cover stories have been used in MLS models for non-key attributes only. None of the existing models has considered a cover story involving a key attribute.

2.1 Covert Channels

The two aforementioned Bell-LaPadula properties prevent the direct flow of information from objects and/or subjects at a higher security clearance level to subjects at a lower level, and are the basis for all MLS models. However, a system may not be secure even if it always enforces the two Bell-LaPadula properties. There may exist a covert channel, which allows for an indirect flow of information from a higher level user to a lower level user. For example, suppose a lower level user wishes to insert a tuple that already exists in the database at a higher level of security. If this insert is rejected by the system, the lower level user will know that there already exists a tuple at a higher level. This indirect flow of information from higher to lower security levels can occur in other ways. For instance, the concurrent execution of transactions results in contention for data objects. If the results from a lower security level transaction are delayed when there is a higher-level security transaction, then the lower security level user can determine there are transactions at higher levels, and may even be able to infer information from the length of the delay.

2.2 Polyinstantiation

In the relational model, two tuples must not exist in a relation with the same values for the primary key attribute, but requiring this constraint to hold in multilevel relations may result in a covert channel. In order to avoid covert channels in MLS data models, subjects with different classifications are allowed to operate on the same relations, through the use of polyinstantiation [7][12]. The term polyinstantiation refers to the simultaneous existence of multiple tuples with the same primary key, where such tuples are distinguished by their classifications [12]. Because of that, the user specified primary key in the MLS environment is called the apparent key. Polyinstantiation is illustrated in Figure 1 where there are 3 tuples representing Diva Megastar.

3. The Key Loophole

Within MLS models the link between a tuple and its corresponding lower-level cover story tuple is the matching value of their entity identifier. In the BCMLS model the entity identifier is composed of the user-defined key and the primary level (pl) of its classification attribute: K + pl(KC). For example, in Figure 1 the second, third and fourth tuples share the same entity identifier (Diva Megastar, U). The entity identifier value is used to identify, on the S level, the second and third tuples as cover stories for the fourth tuple. The same entity identifier identifies on the C level the second tuple as the cover story of the third tuple.

In this section, we will describe the limitation of the BCMLS (and all other existing MLS models) that stems from this definition of the entity identifier. In order to portray this limitation we will slightly alter the example we introduced earlier in this paper.

Example 1 (Altered)

As in the first version of this example, all the information about patient Alan Jones is available for all three security/clearance levels. However, the information about patient Diva Megastar is more sensitive. The subjects on the S level are allowed to see correctly her room number, age, diagnosis, and name. The subjects on the C level are allowed to see her correct room number and diagnosis, but her age and her ***name*** *should be masked by a cover story diagnosis and a cover story name. The subjects on the U level can see her correct room number, and should be given a cover story for her age, diagnosis and* ***name****.*

No existing MLS model is capable of properly handling this scenario. The reason for it is the cover story dependence on the value of a user-defined key. Figure 3 illustrates the situation. If we simply tried to change the name of the patient Diva Megastar to a different name (Julie Smith) on the C and U levels, we will be faced with the following problem. The user on the S level would know that there is no patient named Julie Smith, but at the same time the S user would have no way of knowing that Julie Smith is a cover story for Diva Megastar. Instead, the S level user would treat all records relating to Julie Smith as so-called mirage tuples [10] [11], which represent a non-existing entity (when every attribute of a tuple is labeled as false on a certain level, a user from that level considers that tuple to be a mirage tuple.) This can cause problems in situations when an S level user has to communicate with lower level users. For example the S level user would be unaware that C level users are aware of the patient Diva Megastar (they simply know her under a different name) and her correct diagnosis.

MIDTOWN HOSPITAL PATIENTS TABLE

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
Julie Smith UC-S	Exhaustion U-CS	32 UC-S	201 UC-S	U-CS
Julie Smith UC-S	Intoxication C-S	32 UC-S	201 UC-S	C-S
Diva Megastar S	Intoxication S	42 S	201 S	S

Figure 3

We call the inability of the existing MLS models to connect a tuple that represents a certain entity on a particular security level to a lower-level cover story tuple that has different key attribute value but represents the same entity: *the key loophole*.

The only way existing MLS models can deal with the key loophole is to try to avoid it. For example, in order to avoid dealing with the key loophole in the given example, we could simply keep the S patient completely hidden from the lower level users, as shown in Figure 4.

MIDTOWN HOSPITAL PATIENTS TABLE

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
Diva Megastar S	Intoxication S	42 S	201 S	S

Figure 4

However, this solution restricts access to more information than what was called-for by the requirements: *The subjects on the C level should be allowed to see her correct room number and diagnosis, but her age and name should be masked by a cover story diagnosis and a cover story name. The subjects on the U level should see her correct room number, and should be given a cover story for her age, diagnosis and name.* In addition, this approach has the potential for opening covert channels. For example, unless the patients with sensitive information are kept in a completely separate ward, the information about how many patients are in the hospital may be hard to keep hidden from the lower level users. Therefore, the lower level users can become aware that some information is kept hidden from them and that there is a patient in the hospital that is not included in the database.

4. System Defined Entity Identifier

We introduce a change in the way the entity identifier is defined. We propose a system defined entity identifier (SEID), whose value would remain hidden to all users on all security levels and would be used only internally by the MLS DBMS. We will illustrate how this concept would be used to properly handle the situation depicted in the Example 1. This is shown in Figure 5.

MIDTOWN HOSPITAL PATIENTS TABLE

SEID	Patient Name	Diagnosis	Age	RoomNo	TC
1111	Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
2222	Julie Smith UC-S	Exhaustion U-CS	32 UC-S	201 UCS	U-CS
2222	Julie Smith UC-S	Intoxication CS	32 UC-S	201 UCS	C-S
2222	Diva Megastar S	Intoxication CS	42 S	201 UCS	S

Figure 5

The SEID column contains the new system defined entity identifier. If an S level user requests all information about Diva Megastar, the fourth tuple along with the cover story second and third tuples would be displayed. The S level user would now be aware of the fact Julie Smith's records are cover–stories about Diva Megastar given to the lower level users. An interface to an MLS application can now bundle each tuple with its related cover stories, even if the cover stories are not related via a key value (Figure 6). We will call the cover stories that are not related through a matching value of key attributes – Non Key-related Cover Stories (NKCS).

MIDTOWN HOSPITAL PATIENTS TABLE

Patient Name	Diagnosis	Age	RoomNo	TC
Alan Jones UCS	Exhaustion UCS	56 UCS	101 UCS	UCS
Diva Megastar S	Intoxication CS	42 S	201 UCS	S
Julie Smith UC-S	Intoxication CS	32 UC-S	201 UCS	C-S
Julie Smith UC-S	Exhaustion U-CS	32 UC-S	201 UCS	U-CS
Next Record	…	…		…
Its Cover Stories …	…	…		…

Figure 6

An argument could be made that a simple solution for the key loophole problem is to create and use an attribute Patient ID instead of using the patient name as a key. However, that would not solve the problem. The Patient ID would have to be immutable in order to keep track of a patient's history. If a patient Diva Megastar was ever previously admitted into the hospital without using a cover-story name, using a new name with the same Patient ID would open a covert channel. In other words, the key loophole problem would simply be shifted from the Patient Name attribute to the Patient ID attribute.

In order to enable an implementation of the new model based on the concept of system defined entity identifier, we made the necessary changes to the basic MLS properties and we developed the new relational algebra. We also implemented the proposed model in a prototype application and investigated the performance issues. We found that implementing the new approach can be accomplished without creating a performance overhead. The details of model properties' changes, relational algebra, and performance analysis are beyond the space limit of this paper (but can be found in the tech-report [13]). Instead, we focus on the security improvements that result from the new approach

4.1. Model Robustness Improvements

As we showed already, the existence of the key loophole prevents existing MLS databases from including NKCS, which unnecessarily restricts options for presenting sensitive information. In addition to restricting information presentation options, the key loophole also can pose a security threat. In particular, the key loophole adversely affects two common types of queries: *queries related to physical entities* and *aggregate queries*. Such queries, when involving tuples with classified information in their user-defined keys, cannot be handled in existing MLS model based systems without opening covert-channels.

There are two different ways in which the key loophole can be exploited and consequently covert-channels can be opened. *Outside threat* exists when information that can be gleaned from the real world is inconsistent with information in the database. *Inside threat* occurs when information, that in principle is unclassified and available to users of all security levels, is inconsistent with other information derived from the database. Admittedly, these two types of threats can be related and the distinction between them can sometimes be blurred. Nevertheless, both threats are real and existing MLS models are vulnerable for most practical applications.

Queries related to physical entities are susceptible to outside threats. Intuitively, such queries involve *observable* objects, such as hospital rooms, for which we store information in the database. Thus, the existence of a tuple that has attributes relating to this physical entity can be inferred from observing the real world rather than querying the database. For example, any hospital personnel can notice that a particular hospital room is occupied. Thus, the danger comes not from the database itself but from the information that can be gleaned from the real world. Existing models have not addressed this concern. However, for practical applications, it is crucial that MLS models handle security threats that originate outside the database. We view this consideration and our solution as an important contribution.

For a concrete example, consider the scenario outlined in Example 1. A variety of people working in the Midtown Hospital may notice the fact there is a patient in room 201. Therefore this fact by itself cannot be classified and efforts to obscure or protect it in the physical world will backfire by drawing unwanted attention to it. Instead, our goal is to protect the identity of the patient Diva

Megastar and her diagnosis. Previous MLS models, as discussed in Section 3, can achieve this goal in a way that preserves the integrity of the data only by making the *entire* tuple secret, as shown in Figure 4. This approach however, is susceptible to an outside threat. Consider the following query executed by a user with clearance level U (the lowest, unclassified level).

```
SELECT PatientName
FROM MidtownHospitalPatients
WHERE RoomNo = 201;
```

The result of this query is empty. Thus, there is a glaring inconsistency between the physical world and the database, which opens a covert channel. In contrast, our solution exhibits no such problems. The situation is handled as shown in Figure 5. The result of the above query at clearance level U is the singleton ("Julie Smith"). This result is consistent with the information that can be gleaned from the physical world.

Aggregate queries are susceptible to both outside and inside threats. There are many situations where aggregate information computed from a database is not by itself classified but some of the tuples involved in the aggregation may contain classified information. Furthermore, the result of the aggregation may be obtainable in other ways. Aggregate queries in existing relational models are vulnerable to threats of either type (outside or inside) because of the key loophole - tuples with classified user-defined keys that participate in a given aggregation are not visible at lower clearance levels and thus, do not contribute to the result. In contrast, our model allows invariant aggregate queries that involve classified tuples.

For an example of aggregate queries under an outside threat consider again the scenario outlined in Example 1. In this scenario, the number of patients can be obtained from observing the physical world (e.g. by counting the number of occupied rooms). Therefore, the result of the following query can be corroborated by means outside the database:

```
SELECT COUNT(*)
FROM MidtownHospitalPatients;
```

The result of this query must be the same for all levels of clearance. Any inconsistency opens a covert-channel. The existing models would either have different result for this query on different security levels (thus allowing the outside threat) or would not protect the name of a classified patient. Our approach eliminates this security dilemma.

The next example shows how our model handles a possible inside threat for an aggregate query.

Example 2.
Globreach Corporation keeps track of their various financial accounts in a database. The corporation classifies its database users into three clearance categories: U, C, and S, which determine the sensitivity of information they are allowed to see. The database contains two tables. One table contains information about the accounts and another table contains information about the holder of the accounts. Accounts can have multiple holders. Holders can be various departments and groups within this corporation. Two groups, Central Asia Operation and Africa Operation, share the same expense account X100. This information is available on every security level. The same two groups share another account, multipurpose account T999. The fact that account T999 is held by Central Asia Operation and Africa Operation must be hidden from U and C level users. Cover stories, depicting Development holding account T999 instead of Central Asia Operation, and Miscellaneous Procurement holding account T999 instead of Africa Operation, are created and presented on the C and U levels.

GLOBREACH CORP. FINANCIAL ACCOUNT DATABASE

Table: BANK ACCOUNTS

SEID	AccountNo	Type	Balance	TC
B111	X100 UCS	Expense UCS	$280,500 UCS	UCS
B222	T999 UCS	Multipurpose UCS	$2,330,000 UCS	UCS
B333	X200 UCS	Multipurpose UCS	$432,000 CS	UCS

Table: ACCOUNT HOLDERS

SEID	Account No	Holder	Dept	Last Trans.Date	TC
A111	X100	Cent. Asia Op. UCS	Intl. UCS	5.8.03 UCS	UCS
A222	X100	Africa Op. UCS	Intl. UCS	5.7.03 UCS	UCS
A333	T999	Development UC-S	Intl. UCS	5.5.03 UCS	UC-S
A444	T999	Misc. Proc. UC-S	Intl. UCS	5.5.03 UCS	UC-S
A333	T999	Cent. Asia Op. S	Intl. UCS	5.5.03 UCS	S
A444	T999	Africa Op. S	Intl. UCS	5.5.03 UCS	S
A555	X200	Marketing UCS	Dom UCS	5.2.03 UCS	UCS

Figure 7

In this example, assume that the total budget for the Globreach Co. is available in another table. Thus, it is essential that account T999 is listed in BankAccounts. Otherwise the following query will return a result inconsistent with information available in a different database table.

```
SELECT SUM(Balance)
FROM BankAccounts;
```

One can argue that the information about account T999 is unclassified, and therefore, a previous model can handle this situation. However, the information about the actual account holders must be kept secret and, as a result, the existing models would have to completely hide the S level account holder (no cover story) in the AccountHolders table. On the surface, such a strategy does not represent an immediate violation of the constraints since a referential integrity constraint stipulates that AccountNo in AccountHolders must refer to an AccountNo in BankAccounts and not vice versa. However, accounts with no holders will certainly raise red flags and consequently may open covert channels.

Furthermore, a more complex aggregation that finds the total funds available to different departments will expose the omissions and certainly open a covert channel. The following query computes this aggregation:

```
SELECT H.Department, Sum(A.Balance)
FROM BankAccounts A, AccountHolders H
WHERE A.AccountNo = H.AccountNo
AND H.Holder <= ALL
    (SELECT T.Holder
     FROM AccountHolders T
     WHERE T.AccountNo = H.AccountNo
     AND T.Department = H.Department)
GROUP BY H.Department
```

The sub-query part counts each account only once toward the funds available to a particular department, even if account has more than one holder from a particular department. The result of the whole query for the tables in Figure 7 consists of two tuples: (“International”, 2610500) and (“Domestic”, 432000).

In our model this aggregate query is invariant. Thus, there is no inside threat, even if the same information is stored in another database table with department budgets. In previous models, however, all holders of the T999 account will be kept secret, and thus the above query will produce a *different result* at lower security levels. The result will still consist of two tuples but the sum of the balances available to the International department will be different because the balance in account T999 will not participate in the aggregation.

5. Conclusion

In this paper we identified a major flaw of existing MLS models that restricts their use in practical applications and proposed a functional solution. Our approach considers the interaction between information that can be gleaned from the outside world and the information stored in the database. While this interaction has largely been ignored in the MLS literature, its importance for maintaining the integrity and security of the database is paramount.

The main contributions of this paper are the discovery and analysis of the key loophole in existing MLS models, which serve as the basis for the proposed enhancement of an MLS with system defined entity identifiers. This innovation significantly improves the applicability and the robustness of the model.

References

[1] V. S. Atluri, S. Jajodia, and E. Bertino, "Transaction Processing in Multilevel Secure Databases with Kernelized Architecture: Challenges and Solutions," IEEE Transactions on Knowledge and Data Engineering, Vol. 9, No. 5, pp. 697-708, 1997.

[2] D. E. Bell and L. J. LaPadula, “Secure Computer Systems: Mathematical Foundations and Model,” Technical Report, MITRE Corporation, 1974.

[3] D. E. Denning, “The Sea View Security Model”, Proceedings: IEEE Symposium on Security and Privacy, Oakland, California, pp. 218 -233, 1988.

[4] T. D. Garvey and T. F. Lunt, “Cover stories for database security,” Database Security V: Status and Prospects, edited by S. Jajodia and C. E. Landwehr, North-Holland, 1992.

[5] J. T. Haigh, R. C. O’Brien, and D. J. Thomasen “The LDV Secure Relational DBMS Model,” Database Security IV: Status and Prospects, edited by S. Jajodia and C. E. Landwehr, North-Holland, pp. 265-279, 1991.

[6] Jamil H. M., “Belief Reasoning in MLS Deductive Databases”, Proceedings: ACM SIGMOD, Philadelphia, Pennsylvania, USA, 1999.

[7] S. Jajodia, and R. Sandhu, “Polyinstantiation Integrity in Multilevel Relations,” Proceedings: IEEE Symposium on Security and Privacy, Oakland, California, pp. 104-115, 1990.

[8] S. Jajodia and R. Sandhu, “Toward a Multilevel Secure Relational Data Model,” Proceedings: ACM SIGMOD, Denver, Colorado, pp. 50-59, 1991.

[9] B. Jukic, N. Jukic, L. Meamber, and G. Nezlek, “Implementing Polyinstantiation as a Strategy for Electronic Commerce Customer Relationship Management,” International Journal of Electronic Commerce, Vol. 7, No. 2, pp. 9-30, 2003.

[10] N. Jukic and S. V. Vrbsky, “Asserting Beliefs in MLS Relational Models,” SIGMOD Record, Vol. 26, No. 3, pp. 30-35, 1997.

[11] N. Jukic, Vrbsky S., Parrish A., Dixon B., and Jukic B. “A Belief-Consistent Multilevel Secure Relational Data Model”, Information Systems, Vol. 24, No. 5, pp. 377-402, 1999

[12] T. F. Lunt, Research Directions in Database Security, Springler-Verlag, 1992.

[13] S. Nestorov and N. Jukic, “Implementing SEID as a Solution for Connecting NKCS”, The University of Chicago, Computer Science Department, Technical Report TR-2003-03, 2003

[14] R. Sandhu and F. Chen, “The Multilevel Relational (MLR) Data Model,” Transactions on Information and System Security, Vol. 1, No. 1, 1998.

[15] R. S. Sandhu and S. Jajodia, “Polyinstantiation for cover stories,” Proceedings European Symposium on Research in Computer Security, Toulouse, France, Springer-Verlag, 1992.

[16] K. Smith and M. Winslett, “Entity Modeling in the MLS Relational Model,” Proceedings: 18th VLDB Conference, Vancouver, B.C., pp. 199-210, 1992.

[17] K. P. Smith, B. Blaustein, S. Jajodia and L.A. Notargiacomo, "Correctness Criteria for Multilevel Secure Transactions," IEEE Transactions on Knowledge and Data Engineering, Vol. 8 No. 1, pp. 32 – 45, 1996