CHAPTER 18

Confidentiality vs Integrity in Secure Databases

Adrian Spalka and Armin B. Cremers
Department of Computer Science III, University of Bonn
Roemerstrasse 164, D-53117 Bonn, Germany
adrian@cs. uni-bonn.de

Key words: Databases, Security, Confidentiality, Integrity, Logic

Abstract: The problem of enforcing confidentiality in the presence of integrity constraints in secure and, in particular, in multi level databases is still open. To enforce confidentiality the majority of previous works either advocates a violation of integrity or proposes pragmatically its preservation or restoration. In this work we argue that there can never be a trade-off between these two properties for integrity is a fundamental quality of every database, ie also a secure one. Confidentiality always implies a kind of distortion of the open database. We introduce a formally sound method for its enforcement which relies on aliases, ie, additional tuples the only purpose of which is the preservation of integrity of both the open database and each distortion of it.

1. INTRODUCTION

The two, by far, most controversial issues in the field of secure and multi level databases are:

1. the handling of integrity constraints
2. the relationship of information at different security levels

The actual problems – and the failure to find a formally sound solution – can be best explained by observing the evolution from an open, flat database to a secure or multi level one.

In an open database the fact that a data set is the present database state implies that it satisfies the database's integrity constraints. Formally, integrity constraints represent invariant properties of any database state, ie, they play the role of boundary conditions which are as such given. From a practical viewpoint, a database is expected to be an image of some real-world section. A database state represents a snapshot of this section. The present state is said to be accurate if it truthfully describes the present situation in the section. Apart from the trivial case in which the real-world section is itself static, the database has no means for deciding the present state's accuracy. However, the database is very well able to separate all data sets into two groups: the nonsensical sets, ie those that cannot possibly be accurate, and the potentially accurate, ie the remaining ones. The separation criteria are the integrity constraints. Thus to have an open database with an invalid present state is an unmistakable indicator that the database is definitely not an image of the real-world section. If the integrity constraints are correctly specified, then the state is nonsensical and must be corrected. If the state is accurate, then the integrity constraints are erroneous and must be corrected. In any case, this unsatisfactory situation calls for a corrective action. This seems to be not so if this open database becomes a secure or multi level one.

An open database permits every transaction that leads to a valid state, there are no confidentiality demands and each user has unlimited powers. The integrity constraints' role is interpreted here as a voluntary protection against inadvertent mistakes. At the same time, integrity constraints provide an explanation to a user for a rejected transaction.

In a secure database, we assume that there is a global open database, which captures the open intended state of the world section, and a distorted local database for each user or group of users from which a part of the open state should be kept confidential. Spalka/Cremers (1999) show that a necessary precondition for the enforcement of confidentiality is the determination of a user's powers and an assessment of his assumed knowledge with respect to the database.

Given that a database comprises a signature, a set of integrity constraints and a state, there are at least three factors that can be manipulated in order to satisfy a confidentiality demand. A change in the signature yields a completely different database language, and a change in the integrity constraints affects a lot of database states. Even if successful, these means often incur a too radical distortion of a local database. The least obtrusive encroachment inserts and deletes single elements of the state. These locally distorted elements are called aliases for a confidentiality demand.

When a confidentiality demand is stated, the attempt to enforce it can threaten integrity at two points. Firstly, let us call it the static case, it can immediately violate the local database's integrity. Or secondly, this is the dynamic case, it leaves the local database in a valid state but the users with access to this database can perform a transaction that leaves this local database in a valid state but violates the integrity of the global database, ie, the user's transaction is rejected without a reason that is visible and comprehensible to this user. Given that the integrity of both the global and each local database must be preserved, our alias-based method examines the integrity constraints and the present state and determines if and how a confidentiality demand can be enforced. The distortion of a local database dictated by the confidentiality demand is a necessary one. The use of aliases – which can be both inserted or deleted of tuples – is an additional distortion, which is necessary to enforce a confidentiality demand and, at the same time, to preserve integrity of all databases.

In the static case our method picks the violated instances of the integrity constraints and attempts to modify the local state in such a way, that:

- the violated instance becomes satisfied again (preservation of integrity)
- the modification cannot be modified or deleted by the users of the local database (observation of the users' powers)
- the modification cannot be recognised by the users of the local database as a bogus one (observation of the users' assumed knowledge)

In the dynamic case our method picks the instances of the integrity constraints affected by the confidentiality demand and identifies those instances among them which, after a local user's authorised and valid transaction, remain satisfied in the local database but become violated in the global database, and attempts to modify the local state in such a way that such a transaction will be rejected.

Whenever aliases are needed to enforce a confidentiality demand but no suitable ones can be found, we reject it. The rejection is not a weakness of our method – it is in parallel to the real life situation, where some things can be kept secret and some not due to the person's knowledge and ability.

To make the identification of integrity constraints at threat (ie those that are or can become violated due to a confidentiality demand) tractable, we assume that they have been transformed from a closed formula into a set of normal clauses and possibly a set of rules, which needs to be included into the database state. Such a transformation is always possible[3].

Section 2 comprises quite a detailed and extensive discussion of previous works. The main result, our attempt to enforce confidentiality in the presence of integrity constraints, is presented in chapter 3. The conclusion comments on our approach and mentions some further steps.

2. PREVIOUS AND RELATED WORK

Most algebraic approaches to bring confidentiality in line with integrity are based on SeaView, a multi level relational data model introduced by

[3] Cf, eg, Cremers/Griefahn/Hinze (1994):69.

Denning et al (1987). The reason for classifying single attributes, according to the authors' opinion, is that an element, ie a tuple's attribute value, represents a fact[4] – from the viewpoint of logic a simply wrong statement[5]. In addition to it, a classification is also assigned to the whole relation scheme. The primary-key and the foreign-key constraints are taken to be the only integrity constraints. The authors note that: 'Functional dependencies correspond to real-world constraints; that is, they are not just a property of a particular relation instance'[6] and 'The requirements for consistency [at each security level] affect the integrity rules of the relational model, which are formulated as global constraints on an entire database.'[7] Yet their approach to the handling of multi level integrity does not account for these semantic facts. It relies on purely syntactical adjustments instead. In particular the decision to redefine a multi level relation's primary key as a combination of the standard relation's key and its classification is accompanied by semantic havoc.

Meadows/Jajodia (1987) speak of a conflict between database security and integrity because if there is an integrity constraint defined over data at more than one security level, then a user at a low security level can use accessible data and this constraint to infer information about data at a high security level. In the authors' view one can either upgrade the low data, ie, make them inaccessible to users at the low security level, or enforce the constraint only at the high security level, ie, sacrifice global integrity but preserve data availability.[8] We believe that in the second case the low users could as well stop using the database at all. In their consideration of the primary-key constraint, the authors note that polyinstantiated objects can be a source of ambiguity.[9] But the question 'How do we provide guidelines to the user for choice of the best tuple?'[10] remains unanswered. In general, the proposals are made on the grounds of the opinion that

> ... the more likely it is that integrity can be restored in the future, the more acceptable it is to sacrifice integrity in the present.[11]

An opinion hardly targeted at semantic soundness.

Stachour (1988), Stachour (1989) and Haigh et al (1989) present LDV, an implementation approach of the multi level data model. Although the

[4] Denning et al (1987):220.

[5] A fact, ie a ground atomic formula, has a truth value, but an element is a term and as such has no truth value.

[6] Denning et al (1987):222.

[7] Denning et al (1987):221.

[8] Meadows/Jajodia (1987):95.

[9] In fact, they regard polyinstantiation as a means of reducing ambiguity and not as its origin. Cf Meadows/Jajodia (1987):92.

[10] Meadows/Jajodia (1987):93.

[11] Meadows/Jajodia (1987):98.

implementations of LDV and SeaView differ, the handling of polyinstantiation is the same – increased user flexibility is seen as the solution to semantic ambiguity.[12] In the LDV data model of Haign/O'Brien/Thomsen (1990) security levels are assigned to tuples only. The authors recognise that '... if polyinstantiation is to be meaningful, it must be done in a manner consistent with the semantics of the database. The semantically significant entities in a relational database are tuples and relations'[13] This, admittedly, reduces ambiguity, yet does not eliminate it. The authors also claim that the enforcement of referential integrity and application integrity across access levels '... are problems for which no complete solution is possible'[14]. They suggest to '... establish quotas for each level and then enforce these quotas on a per level basis'[15]. Though possibly over-restrictive, this suggestion is semantically sound.

Gajnak (1988) investigates the adaptability of entity-relationship modelling to multi level security requirements. The author identifies three fundamental principles of multi level databases which must not be violated[16]. The important semantic determinacy principle states that '... factual dependencies should be non-ambiguous'[17]. This property is violated by Sea View's treatment of polyinstantiation. The author gives an example in which polyinstantiation can mean that: one database entry is an alias for another; a secret entry has been leaked; or the two entries refer to two real world objects. He concludes aptly that in this situation referential integrity as such must be ambiguous. Regrettably, the author's final advice – which we strongly support – that '... the determinacy principle should be supported directly by multilevel secure data models'[18] has been given little attention in the following years. Only Burns (1988) argues in direct support of Gajnak (1988) and presents some more examples illustrating the semantic inadequacy of SeaView's handling of polyinstantiation. She realises already that this '... automatic polyinstantiation is in direct conflict with ... the logical consistency of a database'[19].

Following her conviction that the loss of semantics can be as disastrous as the loss of confidentiality, Burns (1990a) undertakes an attempt to define a data model, the referential secrecy model, in which integrity is given the

[12] 'For flexibility, the user should be allowed to specify which tuples are to be filtered away from the response using time-oriented constructs and level-oriented constructs...' Stachour (1988):72.

[13] Haigh/O'Brien/Thomsen (1990):268.

[14] Haigh/O'Brien/Thomsen (1990):266.

[15] Haigh/O'Brien/Thomsen (1990):277.

[16] Gajnak (1988): 189.

[17] Gajnak (1988): 183.

[18] Gajnak (1988):189.

[19] Burns (1988):230.

same priority as confidentiality.[20] The ideas of this rather pragmatic approach are selective limitation of polyinstantiation and selective automatic upgrade of inserted tuples. While clearly not semantically sound, any solution limiting ambiguity should be regarded as an improvement. Burns (1990b) claims that '... the fundamental problem is that either the secrecy of the information within the database can be maintained, or the integrity of the database can be maintained, but not both simultaneously'[21]. While we strongly oppose this claim, we definitely sympathise with the author's opinion that 'Database integrity constraints are fundamentally invariant properties of the state of a database'[22] and with her conclusion that integrity must not be sacrificed. In a pragmatic manner, the author proposes to allow polyinstantiation and to audit it as an error so that the database administrator is able to correct it later.

Several suggestions for resolving the conflict between security and integrity are made in Maimone/Allen (1991). To prevent the duplicate existence of primary keys, the authors propose to prevent a user with a low access class from choosing a key existing at a high access class, eg, by partitioning the key space (a solution also proposed by Jajodia/Sandhu (1991)[23]) or by forcing the user to accept an application generated key – this amounts to a restriction of the database functionality. Referential integrity is not brought in accord with security but simply sacrificed[24]. To correct the resulting inconsistencies, a garbage collection procedure should at some time later remove the dangling tuples. The authors' proposal to replace value integrity constraints with triggers because they '... are procedural and event-based ... [and] say nothing about the current or consistent state of the database'[25] is clearly opposed to any attempt at defining an unambiguous semantics of secure databases.

Sandhu/Jajodia (1993) deal also with referential integrity. This time the authors note that entity polyinstantiation, viz, the existence of two tuples with the same primary key value but different security levels, is responsible for the problem of referential ambiguity. Here one cannot properly determine which foreign key tuples correspond to which primary key tuples. In order to avoid it, the authors simply disallow this kind of polyinstantiation. Again, they suggest to partition the domain of the primary key. At the same time, they present an example in which this measure leads in turn to new problems.

[20] Burns (1990a): 135.

[21] Burns (1990b):37.

[22] Burns (1990b):37.

[23] Jajodia/Sandhu (1991c):70.

[24] 'Our approach is to allow the parent record to be deleted, and to leave the child records in place.' Maimone/Allen (1991):56.

[25] Maimone/Allen (1991):58.

Qian (1994) studies the link between integrity constraints and inference channels in multi level relational databases with tuple-level labelling. Integrity constraints are defined as closed formulae, which also comprise security level specifications. There is a static inference channel if data at a low security level does not satisfy the integrity constraints, and there is a dynamic channel if data with a high security level force an update operation executed by a user with a low security level to be rejected even if the resulting data with the low security level appear to be consistent. Based on a constraint's security level, some formal results on the existence and removal of inference channels are derived. First of all, this approach does not contribute to a sound database semantics since integrity constraints should be considered in the real-world context – here they express properties of the data as such, ie without any security levels. Secondly, to suppress some dynamic channels, the author proposes in a syntactical fashion to accompany updates at a low security level with the insertion of data at a higher security level. Yet she notes[26] that even this move is in general ambiguous – let alone its semantic consequences.

Garvey/Lunt (1991b) show that it is not always practical (possible?) to close an inference channel by upgrading information. They suggest the use of cover stories instead. The authors admit that cover stories will require polyinstantiation of data. They note that

Polyinstantiation raises the issue of correctness of data inferred from information stored at different levels of a database ... Is information inferred by a high user ... from low data contradicted by high data?'[27]

Both the issue and the question are left open.

There are several logic-based works, eg Sicherman/de Jonge/van de Riet (1983), Morgenstern (1987) and Bonatti/Kraus/Subrahmanian (1992). Yet their database model is very simple, in the sense that it lacks the Closed World Assumption, integrity constraints and update operations.

3. INTEGRITY RESPECTING ENFORCEMENT OF CONFIDENTIALITY IN SECURE DATABASES

We use the definition of a database with confidentiality of Spalka/Cremers (1999). To illustrate our approach we make the following assumptions:

- there are three users: u_1, u_2 and u_3
- all users have a common flat name-space, ie we do not need name-space selectors

[26] Qian (1994): 165.

[27] Garvey/Lunt (1991b):377.

- there are three groups of users: $G_1 = \{u_1\}$, $G_2 = \{u_1, u_2\}$ and $G_3 = \{u_1, u_2, u_3\}$
- a database is associated with each group: D_1, D_2, and D_3; D_1 is the open database, ie nothing should be kept secret from u_1, and D_2, and D_3 are distortions of D_1 such that – according to the group members – u_1 can state confidentiality demands for u_2, u_3, and u_2 can state confidentiality demands for u_3.

According to Spalka/Cremers (1999) a successful enforcement of a confidentiality demand is a distortion of the truth value that yields a single distorted model, eg if the tuple α should be kept secret from u_3, then α's truth value in D_3 is opposite to its truth value in D_1.

3.1 Dynamic violation

Let us first have a look at this situation in a general database D and one of its distortions D'. All elements of C, the integrity constraints, are universally quantified clauses, ie a disjunction of literals.

In the dynamic case, the effect of a confidentiality demand stated for a tuple α does not violate any integrity constraint. However, in order to respect the user's powers we must ensure that he cannot execute an authorised transaction, ie one that is within his powers, that is valid in his local database but invalid in the global database.

Given a possible dynamic violation, our idea to enforce confidentiality is to try to ensure that whenever a transaction yields a violated instance of an integrity constraint with respect to D it will also yield a violated instance of an integrity constraint with respect to D'. Each such instance has the form

$$\psi\pi = \eta_1 \vee \ldots \vee \eta_k \vee \varepsilon_1 \vee \ldots \vee \varepsilon_{n-k}$$

such that:

- each η_j is a truthful literal
- each ε_i is a distorted literal
- α or $\neg\alpha$ is among the distorted ε-literals

$\psi\pi$ has the same truth value in D and in D', if k=n, ie all links of $\psi\pi$ are truthful and neither α or $\neg\alpha$ is among them. This trivial case guarantees that this instance of the integrity constraint will always have the same truth value in both the global and local databases, ie a transaction accepted locally will also be accepted globally and if it is rejected by the global database, then it will also be rejected by local database and the user has a local explanation for the failure.

Let $k<n$ and α or $\neg\alpha$ is among the distorted ε-links of $\psi\pi$. The instance $\psi\pi$ is satisfied, ie at least one of its links is true. According to the

observation of powers and knowledge, none of the distorted ε_i is in R_u or K_u. Thus, the effect of every local transaction is limited to the state's truthful, undistorted part, that is, a local transaction can change only the η-links' truth values (simultaneously in both the local and global databases). Therefore, keeping in mind that each ε_i has in D the opposite truth value than in D', if there is at least one ε_i the truth value of which in D' is False, then $\psi\pi$ clearly cannot become false in D while it remains true in D'. Again, integrity is not at threat.

Let us finally turn our attention to the dangerous case. Let $k<n$ and all ε_i be true in D'. The global database's integrity can be threatened if there is a single local transaction (or a sequence thereof) which sets all η_j to False, for then $\psi\pi$ remains true in D' but turns false in D. To prevent this from happening we can, firstly, try to find an η_j we can force to stay true, or, secondly, try to find an instance $\varphi\sigma$ we can force to turn to False in D' whenever $\psi\pi$ turns to False in D. The first case is simple: check if there is an η_j which is true in D' and satisfies also the following conditions:

- η_j is not in K_u: it respects the user's knowledge, ie the user can only learn the truth value of η_j from the database
- η_j is not in R_u: it respects the user's powers, ie the user cannot alter our distortion
- the change of the truth value of η_j does not violate any other integrity constraint

If there is none such literal we must examine the second way. We now need a, not necessarily different, integrity constraint φ and a substitution σ such that:

$\varphi\sigma$ is ground

- $\psi\pi = \eta_1 \vee \ldots \vee \eta_k \vee \xi_1 \vee \ldots \vee \xi_{n-k}$, $\varphi\sigma$ coincides with $\psi\pi$ in the truthful η-links
- all ξ_i are false in D'
- the confidentiality demand for all ξ_i is satisfiable

If we can find such a φ and σ, we make each ξ_i an alias; if we can find more than one combination, pick one or let the user choose one; if there is none, the confidentiality demand for α is not satisfiable.

3.2 Static violation

In the static case, the effect of a confidentiality demand stated for a tuple α violates a subset W of the integrity constraints C, ie, for all ψ in W there is a finite set of substitutions Π_ψ such that for all π in Π_ψ $\psi\pi$ is ground, α or $\neg\alpha$ is among the links of $\psi\pi$ and $\psi\pi$ is false in D', ie violated.

Given a static violation, the idea to enforce confidentiality is to try to change the truth value of each instance $\psi\pi$ in D to True. We are not allowed to change an ε_i's truth value in D' for its distortion is a deliberate consequence of a previous confidentiality demand. Thus we can only achieve our goal by reversing the truth value of an η_j. However, our choice is limited to those η_j which also satisfy the conditions for truth-reversal in the dynamic case.

Each such η_j is called a candidate for an alias, and that η_j actually selected from among all the candidates is called an alias for α and ψ. If the set of candidates is empty, then the confidentiality demand for α is not satisfiable. Otherwise, we have again decided to let the user pick a candidate he considers most appropriate.

3.3 Remark

This theory for secure databases supporting primary key and foreign key constraints is already implemented in a prototype a copy of which can be freely obtained by request to the authors.

4. CONCLUSION

When dealing with confidentiality and integrity in secure databases most previous works took the view that one can or must be traded against the other. We have shown that integrity is a fundamental property of all databases, open and secure ones, and must never be violated or sacrificed. Therefore, whenever a database acquires additional properties or abilities, these must always be introduced in a manner that respects and preserves integrity–there can be no trade-off. Confidentiality is always connected to a distortion of the open database and there are two elements of the database that can be distorted: its signature, viz its scheme, and its state. A distortion of the scheme is a permanent one and, thus, it is useful, whenever the confidentiality demands have a recurring pattern. This work focuses on confidentiality demands which need not be anticipated in advance. Our method to enforce them has two properties. Firstly, whenever the database determines that a confidentiality demand cannot be brought into accord with integrity it is rejected as unsatisfiable. And, secondly, if recognised by the database as satisfiable, our method computes if and how many additional distortions of the state, which we call aliases, are needed to enforce it in a way that preserves present integrity and prevents future violation of integrity. Since this method has a sound formal background, the statements and results can be verified in a rigid proof. As an example, we have demonstrated its application to a primary key constraint. We have a complete proof for its application and limits to primary key and foreign key constraints, although general constraints in databases with rules still lie ahead.

5. REFERENCES

Bonatti, Piero, Sarit Kraus and V.S. Subrahmanian. (1992) 'Declarative Foundations of Secure Deductive Databases'. Ed Joachim Biskup and Richard Hull. *4th International Conference on Database Theory – ICDT'92.* LNCS, vol 646. Berlin, Heidelberg: Springer-Verlag. pp 391-406. [Also in: *IEEE Transactions on Knowledge and Data Engineering* 7.3 (1995):406-422.]

Burns, Rae K. (1988) 'An Application Perspective on DBMS Security Policies'. Ed Teresa F. Lunt. *Research Directions in Database Security.* 1st RADC Database Security Invitational Workshop 1988. New York et al: Springer-Verlag, 1992. pp 227-233.

-----. (1990a) 'Referential Secrecy'. *1990 IEEE Symposium on Research in Security and Privacy.* IEEE Computer Society Press. pp 133-142,

-----. (1990b) 'Integrity and Secrecy: Fundamental Conflicts in the Database Environment'. Ed Bhavani Thuraisingham. *3rd RADC Database Security Workshop 1990.* Bedford, Massachussets: Mitre, 1991. pp 37- 40.

Cremers, Armin B., Ulrike Griefahn and Ralf Hinze. (1994) *Deduktive Datenbanken.* Braunschweig: Vieweg.

Das, Subrata Kumar. (1992) *Deductive Databases and Logic Programming.* Wokingham, England: Addison-Wesley.

Denning, Dorothy E., Teresa F. Lunt, Roger R. Schell, Mark Heckman and William R. Shockley. (1987) 'A Multilevel Relational Data Model'. *1987 IEEE Symposium on Security and Privacy.* IEEE Computer Society Press. pp 220-234.

Gajnak, George E. (1988) 'Some Results from the Entity/Relationship Multilevel Secure DBMS Project'. Ed Teresa F. Lunt. *Research Directions in Database Security.* 1st RADC Database Security Invitational Workshop 1988. New York et al: Springer-Verlag, 1992. pp 173-190.

Garvey, Thomas D., and Teresa F. Lunt. (1991b) 'Cover Stories for Database Security'. Ed Carl E. Landwehr and Sushil Jajodia. *Database Security V.* IFIP WG11.3 Workshop on Database Security 1991. Amsterdam: North-Holland, 1992. pp 363-380.

Haigh, J. Thomas, Richard C. O'Brien and Dan J. Thomsen. (1990) 'The LDV Secure Relational DBMS Model'. Ed Sushil Jajodia and Carl E. Landwehr. *Database Security IV.* IFIP WG11.3 Workshop on Database Security 1990. Amsterdam: North-Holland, 1991. pp 265-279.

-----,-----, Paul D. Stachour and D.L. Toups. (1989) 'The LDV Approach to Database Security'. Ed David L. Spooner and Carl E. Landwehr. *Database Security III.* IFIP WG11.3 Workshop on Database Security 1989. Amsterdam: North-Holland, 1990. pp 323-339.

Jajodia, Sushil, and Ravi S. Sandhu. (1991) 'Enforcing Primary Key Requirements in Multilevel Relations'. Ed Rae K. Burns. *Research Directions in Database Security IV.* 4th RADC Multilevel Database Security Workshop 1991. Bedford, Massachussets: Mitre, 1992. pp 67-73.

Landwehr, Carl E. (1981) 'Formal Models for Computer Security'. *ACM Computing Surveys* 13.3:247-278.

Maimone, Bill, and Richard Alien. (1991) 'Methods for Resolving the Security vs. Integrity Conflict'. Ed Rae K. Burns. *Research Directions in Database Security IV.* 4th RADC Multilevel Database Security Workshop 1991. Bedford, Massachussets: Mitre, 1992. pp 55-59.

Meadows, Catherine, and Sushil Jajodia. (1987) ‘Integrity Versus Security In Multi-Level Secure Databases’. Ed Carl E. Landwehr. *Database Security.* IFIP WG11.3 Initial Meeting 1987. Amsterdam: North-Holland, 1988. pp 89-101.

Morgenstern, Matthew. (1987) ‘Security and Inference in Multilevel Database and Knowledge-Base Systems’. *1987 ACM SIGMOD Conference / SIGMOD Record* 16.3:357-373.

-----. (1988) ‘Controlling Logical Inference in Multilevel Database Systems’. *1988 IEEE Symposium on Security and Privacy.* IEEE Computer Society Press. pp 245-255.

Qian, Xiaolei. (1994) ‘Inference Channel–Free Integrity Constraints in Multilevel Relational Databases'. *1994 IEEE Symposium on Research in Security and Privacy.* IEEE Computer Society Press. pp 158-167.

Reiter, Raymond. (1984) ‘Towards a Logical Reconstruction of Relational Database Theory’. Ed Michael L. Brodie, John Mylopoulos and Joachim W. Schmidt. *On Conceptual Modeling*. New York: Springer-Verlag. pp 191-238.

Sandhu, Ravi S., and Sushil Jajodia. (1993) ‘Referential Integrity in Multilevel Secure Databases'. *16th National Computer Security Conference.* NIST/NCSC. pp 39-52.

Sicherman, George L., Wiebren de Jonge and Reind P. van de Riet. (1983) ‘Answering Queries Without Revealing Secrets’. *ACM Transactions on Database Systems* 8.1:41–59.

Spalka, Adrian, and Armin B. Cremers. (1997) ‘Structured name-spaces in secure databases’. Ed T. Y. Lin and Shelly Qian. *Database Security XI.* IFIP TC11 WG11.3 Conference on Database Security. London at al: Chapman & Hall, 1998. pp 291-306.

-----,-----.(1999) ‘The effect of confidentiality on the structure of databases’. *Database Security XIII.* IFIP TC11 WG11.3 Conference on Database Security.

Stachour, Paul D. (1988) ‘LOCK Data Views'. Ed Teresa F. Lunt. Research Directions in Database Security. *1st RADC Database Security Invitational Workshop 1988.* New York et al: Springer-Verlag, 1992. pp 63-80.

-----. (1989) ‘SCTC Technical Note: Organizing Secure Applications “by Name”'. Ed Teresa F. Lunt. *Research Directions in Database Security II.* 2nd RADC Database Security Workshop 1989. Menlo Park, CA: SRI.

Thuraisingham, Bhavani M. (1991) ‘A Nonmonotonic Typed Multilevel Logic for Multilevel Secure Data/Knowledge Base Management Systems'. *The Computer Security Foundations Workshop IV.* IEEE Computer Society Press. pp 127-138.

-----. (1992) ‘A Nonmonotonic Typed Multilevel Logic for Multilevel Secure Data/Knowledge Base Management Systems – II'. *The Computer Security Foundations Workshop V.* IEEE Computer Society Press. pp 135-146.