DESIGN & DEVELOPMENT OF A MULTILEVEL SECURE DATABASE MANAGEMENT SYSTEM

Mohammed Waheeduddin Hussain1, Prof. P. Premchand2,
Dr. G. Manoj Someswar3
1.Professor & Head, Department of CSE, Nawab Shah Alam Khan College of Engineering & Technology, Affiliated to JNTUH, Malakpet, Hyderabad – 500024, Telangana, India
2.Professor in Department of Computer Science & Engineering, University College of Engineering, Osmania University. Hyderabad-500007, Telangana, India
3.Professor & Dean (Research), Department of CSE, Nawab Shah Alam Khan College of Engineering & Technology, Affiliated to JNTUH, Malakpet, Hyderabad – 500024, Telangana, India

Abstract: The subject matter of Inference problem is basically the problem of users deducing unauthorized information from the legitimate information that they acquire. Our research work particularly concentrates on the inference problem which occurs in a multilevel operating environment. In such an environment, users are cleared at different security levels and they access a multilevel database where the data is classified at different sensitivity levels. A multilevel secure database management system (MLS/DBMS) manages a multilevel database where its users cannot access data to which they are not authorized. However, providing a solution to the inference problem, where users issue multiple requests and consequently infer unauthorized knowledge, is beyond the capability of currently available MLS/DBMSs.

Keywords: Multilevel Secure database management system, Knowledge based Inference Control, Conceptual Graphs, KnowledgeFrames, Rule – based Reasoning, Truth Maintenance System

IJSER

INTRODUCTION

Due to the complexity of the inference problem (see for example [THUR90al), we believe that a triple approach to research to research is needed to combat it; one is to build inference controllers which act during transaction processing, the other is to build inference controllers for database design, and the third is to build inference controllers to act as advisors to the System Security Officer (SSO). In our research paper, we have described prototypes for handling the inference problem during query and update processing [FORD90, COLL90]. In addition, techniques for handling this problem during database design have also been proposed [THUR9la]. While the previous approaches enable the detection and/or prevention of simple inference strategies that users could utilize to draw inferences, we believe that for an inference controller to be effective, it should be able to capture the complex reasoning strategies of humans. In other words, what is needed is a knowledge-based inference controller.

Knowledge-based inference control is a two-step process. The first step is to represent the multilevel application as completely and accurately as possible. The second step is to reason about the application so that security violations via inference could be prevented and/or detected. In this research paper, we discuss the use of conceptual graphs for representing the multilevel application. A tool based on conceptual graphs could be utilized by the SSO to design the multilevel database application. While the computation techniques developed for conceptual graphs could be utilized for reasoning about the multilevel database application, the output from the MLS/DBMS also plays a significant role in users making unauthorized deductions. This means that any reasoning tool must also take into consideration the responses released by the MLS/DBMS and audit data in order to effectively prevent/detect security violations via inference. In section 3 of this paper we discuss the essential points towards designing such a tool. Figure 1 illustrates the two step process involved in knowledge-based inference control. We envisage that a tool based on the approach described here could be utilized by the SSO to detect/prevent security violations via inference. The front-end of the tool represents the multilevel database application, responses released by the MLS/DBMS, and the audit data in a format that can be understood by the SSO. The back-end of the tool reasons with the knowledge and detects/prevents certain security violations via inference.

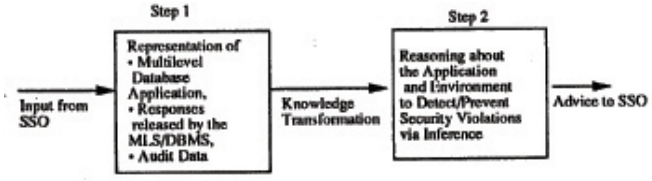

Figure 1. Knowledge-based Inference Control

REPRESENTING AND REASONING ABOUT MULTILEVEL DATABASE APPLICATIONS

We have utilized conceptual structures for representing and reasoning about multilevel database

applications. In particular, we have examined the use of semantic nets as well as conceptual graphs for this purpose. The use of conceptual structures for inference control was first proposed by Hinke [HINK88] where the use of graph theoretic techniques was described. Later Smith [SMIT90] investigated the use of semantic data models for representing multilevel applications. The work reported in [BUC89] also investigated the use of semantic data modeling techniques for controlling inferences in a multilevel environment. The use of conceptual graphs to handle the inference problem was first introduced in [THUR90b] and later in [HINK92]. Other work on the use of conceptual structures for representing and/or reasoning about multilevel database applications is reported in [BINN92, GARV92, SELL92].

Among the various conceptual structures such as semantic nets, semantic data models, and conceptual graphs, conceptual graphs seem to be the most appropriate scheme for representing complex applications. This is because conceptual graphs subsume other structures such as semantic nets and they have the full power of first order logic. [1] Unlike logic-based systems, conceptual graphs represent knowledge in a manner similar to the way humans view the world. Furthermore, they can also be extended to include modality and time without much difficulty. Another advantage of using such a scheme is that the techniques developed for reasoning with conceptual graphs could be utilized for detecting security violations via inference (see for example the discussion in [SOWA84l].

We have chosen conceptual graphs for representing multilevel database applications. Although reasoning with conceptual graphs is as powerful as reasoning with a logic programming system, most of the current knowledge-based systems are not based on conceptual graphs. Therefore, in our approach, the back-end of the inference controller, shown in figure 1, reasons with knowledge represented in the form of rules and frames. In other words, the conceptual graph representation utilized by the front-end of the inference controller must be transformed into frames and rules in order to be processed by the back-end. The use of conceptual graphs is described in section 2.2. The back-end of the inference controller is described in section 3.

DESIGN & DEVELOPMENT

CONCEPTUAL GRAPHS

The use of conceptual graphs for handling the inference problem was first proposed in [THUR90b]. However, in [THUR90b], the use of inference rules for conceptual graphs to detect security violations via inference was not addressed. In this research paper, we review some of the essential points in conceptual graphs for representing multilevel database applications, and discuss with an example how security violations may be detected.

As stated in [SOWA84], a conceptual graph is a finite connected bipartite graph which consists of concepts and conceptual relations. Every conceptual relation has one or more arcs, each of which is linked to a concept. We define a multilevel conceptual graph to be a conceptual graph in which some of the concepts and conceptual relations are sensitive. Figure 2 shows a multilevel conceptual graph (which was represented using a semantic net in [THUR90bl). The Unclassified interpretation of this graph is as follows: CHAMPION carries passengers. Its captain is Smith who has 20 years experience. The ship is located in the Mediterranean Sea on 16 June 1990. It's destination is Greece. The Secret interpretation is as follows: CHAMPION carries SPARK which is an explosive. Its captain is Smith who has battle management experience. The ship is located in the Mediterranean Sea on 16 June 1990. Its destination is Libya. (Note that the Secret concepts and relations are illustrated by darkened structures and lines.) [2]

In [THUR90b], some formation rules (for example, the join of two conceptual graphs, adding connectives such as negation to a conceptual graph) were discussed. These formation rules produce new conceptual graphs. However, these formation rules do not enable any computation. In order to detect security violation via inference, some form of computation with conceptual graphs needs to be performed. In [SOWA84], several types of rules of inference have been proposed for conceptual graphs. These rules enable computation with conceptual graphs. Figure 3 illustrates a deduction rule similar to Modus Ponens in logic. Figure 3(a) illustrates at the unclassified level the fact that if CHAMPION is sailing to Libya, then it must be a warship. Figure 3(b) illustrates at the Secret level the fact CHAMPION is a warship and at the Unclassified level the fact that it is a passenger ship. Figure 3c illustrates at the Unclassified level the fact that Champion is sailing to Libya. The set of graphs shown in figure 3 is inconsistent as there is contradictory information at the Unclassified level. If a set of graphs is inconsistent, then there is a potential for a security violation via inference.

IJSER

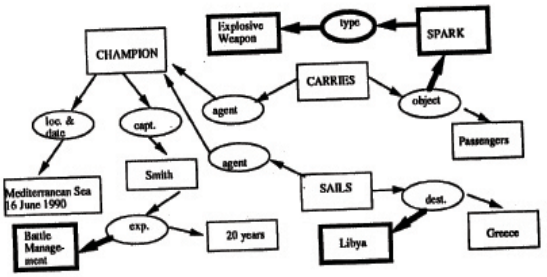

Figure 2. Multilevel Conceptual Graph

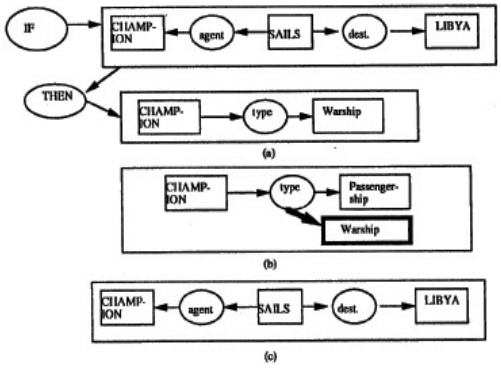

Figure 3. Inconsistent Set of Conceptual Graphs

KNOWLEDGE-BASED INFERENCE CONTROL

In this research paper, we discuss the issues involved in designing the back-end of the inference controller illustrated in Figure 1. We will call this module the knowledge-based inference controller(KBIC). Our research work also describes the modules of the system.

MODULES

The major modules of the KBIC are shown in Figure 4. They are: the User Interface (UI), the Knowledge Manager (KM), the Inference Engine (IE), the Conflict/Contention Resolution System (CCRS), and the Truth Maintenance System (TMS), A description of each module is given below:

UI is the interface to the KBIC. It can be used for updating the knowledge base, for querying, for obtaining advice from the KBIC, or for requesting the KBIC to solve a particular problem. UI is also used if additional information is required from the SSO. Furthermore, UI is the module which interfaces to the tool which is used to represent the multilevel database application. KM is responsible for managing and structuring the knowledge base. It must also ensure the consistency of the knowledge base. Any access to the knowledge base is via KM. It has interfaces to all of the modules of the KBIC. The knowledge base stores all of the relevant information. This includes security constraints, real-world information, heuristics, and relevant information released to various users.[3] IE is the heart of the KBIC. It has the potential for using a variety of inference strategies. As a minimum, IE should be able to perform logical inferences. Note that in a multilevel environment, there could be different views of the same entity at different security levels. This means that the knowledge base could potentially have conflicting information about an entity at conceptually different security levels. Therefore IE should be able to reason across security levels. CCRS is responsible for resolving conflicts as well as determining, the best choice to take when the system is presented with different options. For example, one particular reasoning strategy could potentially give results which conflict with another reasoning strategy. In such a situation, IE would consult CCRS to resolve the conflict. The conflict is resolved by CCRS querying either the KM or even the SSO if necessary. TMS is the module that is responsible for maintaining the consistency of the various beliefs. Such a module is necessary for nonmonotonic reasoning.

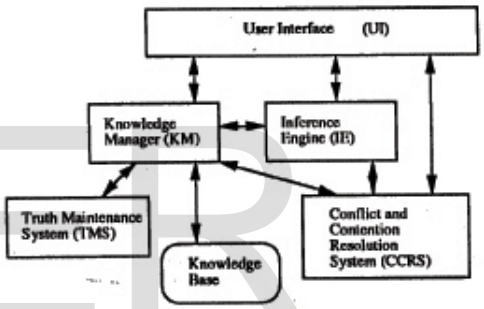

Figure 4. Modules of the KBIC

KNOWLEDGE REPRESENTATION SCHEME

The knowledge representation scheme used by the KBIC is a combination of frames and rules. Frames are ideal to represent structured knowledge. The inheritance mechanism in frames is a powerful one which enables the representation of generic entities, as well as instantiations of the generic entities. The frames used to represent the knowledge are called knowledge frames. Each knowledge frame describes a generic entity or a specific instance of a generic entity. A knowledge frame has many slots associated with it. Each slot describes some property of the entity represented or it could have rules or security constraints associated with it.[4]

Every knowledge frame has one slot which specifies the security level at which the knowledge frame is true. Furthermore, since users at different levels could have different views of the same entity, frames at different levels are used to represent such views.[5] Figure 5 shows Unclassified and Secret knowledge frames which have information on the ship CHAMPION. Since CHAMPION is a ship, it inherits information from the knowledge frame which has information on the generic entity SHIP. Each

knowledge frame also has real-world information and security constraints associated with it. Note that whenever the word "inherit" is used for a slot, it means that the value for that slot is inherited from the knowledge frame representing the generic entity of the specific instance.

Name of Entity : Smith
Entity Type : Captain
Security Level : Unclassified
Skills : 20 Years Experience

Name of Entity : Smith
Entity Type : Weapon
Security Level : Secret
Weapon Type : Explosive

Name of Entity : Spark
Entity Type : Weapon
Security Level : Secret
Weapon Type : Explosive

Name of Entity : SHIP
Entity Type : Generic
Security Level : Unclassified
Information in Database : Ship, Ship-name, Mission
Other Information : None
Security Constraints : None
Instances : CHAMPION

Name of Entity : CHAMPION;
Entity Type : Instance of SHIP
Security Level : Unclassified
Information in Database : Inherit
Other Information :
(i) The destination is Greece
(ii) If destination is Libya there will be war
(iii) If ship is in the Pacifie, then it cannot go to Libya
(iv) Inherit
Security Constraints : If destination is Libya then all mission related information of CHAMPTION is Secret

Liby

Secr

Name of Entity : CHAMPION;
Entity Type : Instance of SHIP
Security Level : Secret
Information in Database : Inherit
Other Information :
(i) The destination is Libya
(ii) If destination is Libya there will be war
(iii) If ship is in the Pacific, then it cannot go to Libya
(iv) Inherit

Security Constraints : Inherit

Figure 5: Knowledge Frames

Name of Entity : CHAMPION
Entity Type : SHIP
Security Level : Unclassified
Location : Mediterranean Sea
Date : June 16, 1990
Destination : Greece
Carries : Passengers
Captain : Smith

Name of Entity : CHAMPION
Entity Type : SHIP
Security Level : Secret
Location : Mediterranean Sea
Date : June 16, 1990
Destination : Libya
Carries : Spark
Captain : Smith

Figure 6: Transformed Knowledge Frames

In addition to representing knowledge as frames, rules are also used to represent some of the knowledge such as constraints, real-world data, and conflict resolution. The rules could be specified in a logic-based language such as datalog [ULLM88].

Representing the multilevel database applications as well as the input from the MLS/DBMS in the form of frames and rules may not be straightforward for complex applications. Therefore, representing the application first using conceptual structures such a conceptual graphs will ease the burden placed o the knowledge engineer. The tool which represents the multilevel database application described in section 2 bridges the semantic gap between the world and the knowledge base. Furthermore, tools have been developed to transform applications represented using conceptual structures into frames and rules [SOWA 84]. In Figure 6, we show how the graph of Figure 2 which may be represented as a collection of frames.

RULE – BASED REASONING

The KBIC uses rule-based reasoning and frame-based reasoning. In addition, it also reasons across security levels. Some of the essential points are discussed in this section. In order for the inference controller to be effective it must also reasoning under uncertainty and utilize additional inference strategics such as inductive and heuristic reasoning. Such reasoning techniques will be part of the future investigation.

Rule-based reasoning techniques include forwards chaining, backward chaining, and hybrid approaches. We illustrate how security violations via inference may

be detected with a simple example. Consider an Unclassified rule base consisting of the following two rules:

R1: CHAMPION is a warship
R2: If X is a warship, its mission is Secret
R3: CHAMPION's mission is Iraq Crisis

Suppose an unclassified user is given the information R1 and R2. Then this release of information must also be recorded in the knowledge base (by KM). IE could reason as follows: since CHAMPION is a warship, using rule R2, its mission is Secret. Since CHAMPION's mission is Iraq crisis, this mission must be kept Secret. Since CHAMPION's mission has been given to an Unclassified user, a security violation has occurred.[6]

As stated in [FROS86], the problem solving technique used by frame-based systems is "matching." Given some information about an entity in the real world, the system will try to match the values associated with the entity with the solt values of frames. We illustrate how security violations via frame-based inference could occur with a simple example. Consider an Unlcassified frame which describes all of the properties of a passenger ship named CHAMPION. Suppose OHIO is another ship and there is a security constraint that classifies all properties of OHIO at the Secret level. There is also an Unclassified rule which states that OHIO and CHAMPION are similar. From this rule, an Unclassified user could infer some of the Secret properties of OHIO. Therefore, one should classify the fat that OHIO and CHAMPION are similar at least at the Secret level.[7]

IE should be able to reason across security levels. In the example of Figure 5, when IE is reasoning at the Unlcassified level (i.e. to detect/prevent unauthorized inferences that users at the Unclassified level could make) it considers the knowledge frame on CHAMPION at Unclassified level. If it is reasoning at the Confidential level, then it still considers the knowledge frame at the Unlcassified level, as thee is no knowledge frame on CHAMION at the Confidential level. If it is reasoning at the Secret level, then it could do one of the following:

- Consider only the knowledge frame on CHAMPION at the Secret level.
- Consider both the knowledge frames n CHAMPION at the Unclassified and Secret levels.
- Consult with CCRS as to which frame to consider.

A simple solution would be to take the first action, That is, assume that information at level L is more accurate than the information at level L-1. In reality, however, information at a lower level could be more accurate. For example, information at a lower level could be more current than the one at the higher level. CCRS could resolve the conflicts either by (i) checking the knowledge base for appropriate conflict resolution rule, (ii) querying the user to give more up-to-date information, (iii) in the absence of appropriate information, make heuristic guesses based on recent experiences, and (iv) reason using the rules of a theory such as plausibility theory [FROS86].[8]

ISSUES ON TRUTH MAINTENANCE STSTEM

TMS is the module of the KBIC that is responsible for maintaining the consistency of the various beliefs. Such a module is necessary for nonmonotonic reasoning. In this section we discuss the essential points in extending Doyle's Truth Maintenance System (TMS) [DOYL82] to reason in a multilevel environment.[9]

In TMS, statements of belief ae called 'nodes.' Each node (or statement of belief) is assigned a security level. If a node is assigned a security level L, then it can be IN or OUT with respect to any level > L. A node is IN with respect to L if it is believed to be true at L. Otherwise, the node is OUT. Each node at level L has a set of justifications linked to it with respect to each security level that dominates L. Each justification at a level L * > L represents a justification representing one way in which the node (i.e., the belief which corresponds to it) may be true. If a justification at level L * is valid, then, unless that justification is explicitly made in valid at level L** (L** is the least level which dominates L*), it is also assumed valid at level L**. A node at level L is IN with respect to level L*>L if it has at least one justification valid at L*. If all justifications at level L* are not valid, then the node is OUT with respect to L*.

We illustrate the essential points of a truth maintenance with an example. In this example, we assume that there are only two security levels, Unclassified (U) and Secret (S). Figure 7 shows the TMS nodes and justifications at the Unlcassified level. This figure is interpreted as follows. The nodes are numbers 1 through 4. Each node has the following assertion or belief. Node 1 has the assertion "CHAMPION is a ship." This assertion has the status IN and does not have any justifications associated with it. Node 2 has the belief "CHAMPION sails to Japan." In order for this belief to be IN, node 1 must be IN and node 3 must be OUT. Node 1 is IN. We will see that node 3 is OUT. Therefore, Node 2 is IN. That is, CHAMPION sails to Japan is consistent with everything that is believed with respect to the Unlcassified level. Node 3 has the belief "CHAMPION is not a passenger ship." In order for this belief to be true, node 4 must be IN. We will see that node 4 is OUT. Therefore, Node 3 is OUT. Node 4 is a previous assertion "CHAMPION carries explosives." It has the status OUT because it must have been retracted earlier.

	Justification

IJSER

Node	Status	IN	OUT
1. Champion is a ship	IN		
2. Champion sails to Japan	IN	1	3
3. Champion is not a passenger ship	OUT	4	
4. Champion carries explosives	OUT		

Figure 7: Justifications at the Unclassified Level

		Justification1		Justification2	
Node	Status	IN	OUT	IN	OUT
1. Champion is a ship	IN				
2. Champion sails to Japan	OUT	1	3		
3. Champion is not a passenger ship	IN	4		5	
4. Champion carries explosives	OUT				
5. Champion is a warship	IN				

Figure 8: Justifications at the Secret Level

Figure 8 shows the assertions, beliefs, and justifications at the Secret level. Here there are two justifications that could possibly be associated with a node. This table is interpreted as follows. There are four unclassified nodes (i.e. beliefs) as in the Unclassified world and one Secret node. Node 1 has the assertion "CHAMPION is a ship." This assertion has the status IN and does not have any justifications associated with it. Note that node 1 is assigned the Unclassified level. Its status has not changed from the Unclassified world. Node 2 has the belief "CHAMPION sails to Japan." In order for this belief to be IN, node 1 must be IN and node 3 must be OUT. Node 1 is IN. We will see that node 3 is also IN. Therefore, Node 2 is OUT. That is , CHAMPION sails to Japan is not consistent with everything that is believed with respect to the Secret level. Note that node 2 is assigned the Unclassified level. Its status has changed from the Unclassified world.[10] Node 3 has the belief "CHAMPION is not a passenger ship." In order for this belief to be true, either node 4 must be IN or node 5 must be IN. We will see that node 5 is IN. Therefore, Node 3 is IN. That is, "CHAMPION is not a passenger ship" is consistent with everything that is believed with respect to the Secret level. Note that node 3 is assigned the Unclassified level. Its status has changed from the Unclassified world. Node 4 is a previous assertion "CHAMPION carries explosives." It has the status OUT because it must have been retracted carlier. Note that node 4 is assigned the Unclassified level. Its status has not changed from the Unclassified world. Node 5 is an assertion "CHAMPION is a warship." It has the status IN. Note that node 5 is assigned the Secret level and is, therefore, not visible at the Unlcassified level.

If at a later time the assertion that "CHAMPION is a warship" is retracted in the Secret world, then the status of node 5 becomes OUT. This would change the status of node 3 to be OUT. The world, in turn, change the status of node 2 to be IN. It should also be noted that a TMS does not create justifications. The justifications are provided to KM to TMS. The TMS maintains a consistent set of beliefs with respect to all security levels.[11]

RESULTS & IMPLEMENTATION ISSUES

One of the ways to implement the KBIC would be to use an existing expert system shell. Commerical off-the-shelf expert system shells such as G2 (product of Gensym Inc.) are now available. Many of these shells handle knowledge bases represented as frames and rules. While using a commercial shell has obvious advantages, such as reduced implementation time and effort, it may not be tailored to solve special problems. That is, one has to contend with the reasoning strategies implemented by the inference engine of the shell. Any additions and/or enchancements to the reasoning strategies may be quite complex to implement. Also, one would need the source code of the shell to make these enhancements. Therefore, unless we can find a shell that can specifically handle the reasoning strategies of the KBIC, this may not be a desired approach. Another approach is to implement the KBIC in a conventional language such as C. While implementation in C has obvious advantages with respect to efficiency, some of the complex reasoning strategies and data structures may be difficult to implement.

A third approach is to use an AI language such as Lisp or Prolog. While both languages have their advantages and disadvantages, since we are mainly interested in handling the inference problem in a relational database management system, the preferred language seems to be Prolog. This because there is a natural relationship between the Prolog data model and

IJSER

the relational data model. In fact, a relational database is a Prolog program [LLOY87]. Prolog interfaces to relational database systems are increasing [LI84, ICOT87]. Furthermore, all of the essential features of the KBIC, such as reasoning under uncertainty, truth maintenance, and handling frame and rule-based representations, can be implemented in Prolog (see for example the discussion in [MERR89]). For these reasons, Prolog may be an appropriate language to implement the KBIC.

CONCLUSION AND FUTURE SCOPE

In this research paper, we have described the inference problem in multilevel database management systems, identified the needs for knowledge-based inference control, and discussed the issues involved in developing a knowledge-based inference controller. Building a knowledge-based inference controller is a two-step process. The first step is to represent the multilevel database application. The second step is to develop techniques for reasoning about the application. We first proposed the use of conceptual structures, such as conceptual graphs, for representing the application. Such a scheme was proposed as it was a natural way to model the world and it had the full power of first order logic. Then we described the essential points of the module which reasons with the knowledge represented in the from of frames and rules. In order for the inference controller to function effectively, the knowledge represented as a collection of conceptual graphs must be transformed into frames and rules.

The developments in artificial intelligence techniques show much promise for the design and development of inference controllers. There is still much work to be done on knowledge representation, knowledge transformation, reasoning under uncertain and incomplete information, and handling different types of inference strategies that users could utilize to draw unauthorized inferences.

REFERENCES

1. Binn,s L., August, 1992, inference Through Secondary Path Analysis," Proceedings of the 6th IFIP Working Conference in Database Security, Vancouver, British Columbia.
2. Buczkowski, L. J., and E. L. Perry, February 1989, Database Inference Controller, Interim Technical Report, Ford Aerospace Corporation.
3. Collins, M., October 1990, Design and Implementation of a Secure Update Processor, Technical Report MTR10977. The MITRE Corporation (a version published in the Proceedings of the 7th Computer Security Applications Conference – coauthors: W.Ford and B.Thuraisingham).
4. Doyle, J., 1982, "A Truth Maintenance System," Artificial Intelligence Journal, Vol.12.
5. Ford , W.R., J. O'Keeffe, and B. Thuraisingham, August 1990, Database Inference Controller: An Overview, Technical Report MTR 10963 Vol. 1, The MITRE Corporation.
6. Frost R., 1986, Introduction to Knowledge-Base Management Systems, Collins, London.
7. Garvey, T., et al, August 1992, Toward a Tool to Detect and Eliminate Inference Problems," Proceedings of the 6th IFIP Working Conference in Database Security, Vancouver, British Columbia.
8. Hinke, T., April 1988, "Inference Aggregation Detection in Database Management Systems," Proceedings of the IEEE Symposium on Security and Privacy.
9. Hinke T., and H. Delugach, August 1992, "Aerie: An Inference Modeling and Detection Approach for Databases, "Proceedings of the 6th IFIP Working Conference in Database Security, Vancourver, British Columbia.
10. "ICOT Project," 1987, New Generation Computing Journal, Vol.5
11. lenat, D., and R.Guha, 1989," Building Large Knowledge-Based Systems," Addison Wesley, MA.

IJSER