Development of a Concurrency Control Technique for Multilevel Secure Databases

Pooja Sapra
Research Scholar, MRIU
mrs.sapra@gmail.com

Suresh Kumar
FET, MRIU
enthuvs@gmail.com

Abstract— ***Multilevel secure database systems are the systems in which security classifications are assigned from the relations to data elements. Due to security requirements of databases, the concurrency control mechanisms for such databases are different than the concurrency control mechanisms in traditional databases. In this paper, we present a new algorithm for concurrency control that is shown to be starvation- free to some extent.***

Keywords - multilevel secure databases, concurrency control, covert channel.

I. INTRODUCTION

As the need of database is increasing, so as the need of securing the data is also increasing. Multilevel secure databases are shared by local databases and users at different sites and the transaction processing takes place at different security levels. So the concurrency requirements include the secure access of data items that are shared by different transactions. These databases are based on the Bell-LaPadula model [17] and consist of a set of subjects, objects, an access control matrix and security levels. Subjects can be user and processes while the objects are the data items or fields on which the access is required. For the information flow it enforces the following two restrictions: simple security restriction and star property restriction.

Simple Security Property: A subject can have a read access on an object if the clearance level of the subject is identical to or higher than the classification level of the object.
Star Property: A subject can write on an object if clearance level of subject is identical to classification level of the object.

Due to these two restrictions on information flow, the transaction can't be executed concurrently with the same mechanism as with traditional mechanisms. Moreover, the present protocols [2, 3, 4, 5, 6, 9, 10, 11, 12] leave the high level secure transaction in a waiting state unless the low level secure transaction commits, it is known as starvation. We therefore present a starvation free concurrency control mechanism for multilevel secure databases that does not suffer from covert channel and retrieval anomaly. The paper is organized as follows: In section 2, we consider the requirements of concurrency control mechanism in secure databases vs. traditional databases. Section 3 is devoted to recent relevant literature. In section 4, we present the proposed algorithm. Section 5 illustrates the performance of the algorithm on securedatabase systems. Finally in section 6, we conclude the paper with some observation about overheads in secure databases.

II. CONCURRENCY CONTROL MECHANISM IN TRADITIONAL DATABASES VS. SECURE DATABASES

The basic conditions of concurrency control include the basic transaction processing requirements, namely, atomicity, consistency, isolation and durability. Due to the information flow model given by Bell Lapadula[17], the direct information flow from high security level to low security level is prevented because of the two properties given in the model. But, there may exist some indirect flow of information due to covert channels. Covert channels are the channels that are established during the information flow without being in the ones' knowledge.
This lead to the additional requirements of concurrency control mechanisms in secure databases, which are given by N.Kaur et.al.[19] as follows:

- Confidentiality Requirements: It must be free of signaling channels
- Integrity Requirements: It must guarantee serializability.
- Availability Requirements: There should be no infinite delays or repeated aborts, means no starvation.

III. LITERATURE REVIEW

Various techniques have been proposed in the literature for concurrency control mechanisms in multilevel secure relations. A multilevel system is the system in which users can access the data according to the classifications and security levels of data. Multilevel secure systems are trusted because they store data with different levels of security and ensure that the properties, simple security and (strong) star are enforced as suggested by Bell LaPadula model. A considerable attention has been paid to research and development of multilevel secure databases.Several approaches have been proposed for concurrency control in MLS/DBMSs. These approaches extend the 2PL protocol or time-stamp based protocol. However, all of these protocols suffer from problems like deadlocks, covert channel, delay or starvation of high security level transactions, and retrieval anomaly. Jajodia and McCollum [3], proposed a secure locking-based protocol S2PL that was based on the strict two phases locking protocol. According to this protocol, when a write lock request is sent by a low security level transaction on a data item, then high security level transaction releases the lock on that data item. This protocol eliminates the problems of covert channel and integrity, but leads to starvation.

McDermott and Jajodia [4] presented a protocol to reduce the starvation. In this protocol, high level transaction does not roll back and aborts completely. However, it holds write locks on high security level data. This protocol does not guarantee the serializable schedules.

Son and David [6], proposed a secure two-phase locking-based protocol (S2PL) on the basis of the concepts of virtual locks. Virtual locks are used by low security level transactions and are implemented on their own versions of the data item. Virtual locks may be upgraded to a real lock, when the highly secure transaction commits and releases the data item.

E. Bertino et al. [7], presented a protocol that uses data items with single-version. This approach was based on nested transactions, notification-based locking protocols and application-level recovery. This approach satisfies all properties given by Atluri et al. [2] and Kaur et.al.[19].

IV. PROPOSED CONCURRENCY CONTROL ALGORITHM

The proposed algorithm is an effort to achieve the concurrent execution of transactions without leading to starvation. The algorithm is based on multi-version concurrency control protocols. In this algorithm version locks are attached to remove the retrieval anomaly and a logical degradation of locks is considered to reduce the priority of low level transaction and so as to increase the high level transaction's priority. This degradation leads to limit the starvation to some extent and also eliminates the covert channel.

DEFINITION 1: The conflict data-items are stored in c-data-items. This data structure contains the set of data-items, read by high level secure transaction and a new version of these data-items is written by low level secure transactions.

C-DATA-ITEMS: {READ-SET-DONE (Ti) ∩ WRITE-SET (Ti+1)} where, READ-SET-DONE(Ti) consists of the data items that have been read in transaction Ti, before the starting of transaction Ti+1, and WRITE-SET(Ti+1) stands for the data items that are to be written by the new transaction.

DEFINITION 2: v-lock denotes the version lock. So, when a data item is to be written, the new version of data item is created. As soon as a new version is created, it is v-locked so that the latest version is available for reading and updation.

DEFINITION 3: Further, the function LDeg is used to virtually degrade the level of transactions. High level transaction is automatically degraded by one level logically as and when a low level transaction interrupts a high level transaction. The degraded levels are stored in the variable VL, virtual levels. AL, stores the values of actual level and that is equal to the level of transaction.

For instance if AL = high; then VL = low, assuming the fact that high>low.

The algorithm for concurrency control is as follows:

```
Step 1:  initially at t = 0;
         c-data-item = null;
         v-lock = null;
At time t=0 for transaction Ti:  VL(Ti,0)=AL;
Step 2:  Repeat until no more transaction in scheduling queue:
When Ti is executing and Ti+1 comes for execution, following three cases may arise:

    (a)    L(Ti) < L(Ti+1)
    (b)    L(Ti) = L(Ti+1)
    (c)    L(Ti) > L(Ti+1)

In case of (a) and (b) normal execution will take place, whereas in the third case execution would involve the following steps:
(i)      Find c-data-item = read-set-done(Ti) intersection
         write-set(Ti+1)
(ii)     Put v-lock on all items in c-data-item.
(iii)    Find the virtual level for all transactions using the
         function: VL=LDeg(VL) Where, LDeg is a function
         that degrades the level of transaction logically by one
         level so as to avoid the starvation.
(iv)     Execute the transaction with the lowest value of virtual
         level, if two transactions have same virtual levels then
         compare the actual level values and execute the
         transaction with the highest level of security.
Step 3:  While resuming the execution of transaction Ti, if it
         has a data item with v-lock then roll back the
         transaction and execute with latest version of data
         item. According to the algorithm the retrieval anomaly
         will be totally eliminated. However, for making the
         system starvation free, Ldeg function is used.
```

When the value of virtuallock becomes unclasified(the lowest security level), it starts executing the corresponding transaction rather than the transaction whose actual lock value is low. In this way, it may introduce some covert channel. So there may be a tradeoff between the covert channels established and starvation.

V. ILLUSTRATIVE EXAMPLES

A. Number of security levels=2, Number of transactions=3

Let us consider the case of 2 levels for security classifications of data, with the partial ordering relation as follows:

HIGH (H) > LOW (L)

The transactions T1, T2 and T3 are issued by high, low and high security level users, respectively.

According to the algorithm their execution is shown in figure 1.

Initially, T1 reads a low level data item A, the transaction T2 also wants to read that and then wants the write access on that. To avoid the covert channel, Bell-Lapadula model gives the priority to execution of low level secure transaction. But this may lead to starvation of high level secure transaction, if all the upcoming transactions are low level secure. The proposed algorithm works as follows: When T1 is executing and T2 enters, then c-data-item set is found. In this example this set contains A and also a new version of data item is created when T2 wants to write on data item A. So it is version locked.The execution of this scenario is also represented in table 1. In this table each entry has a triplet (Virtual level, c-data item set, v-lock). Initially at time t=1 :

VL (Virtual Level) = AL (Actual Level) :
c-data item set = null :
v-lock = null :

Therefore at t=1, the entry in the table for transaction T1(H) is (H, null, null).

At time t = 2, when transaction T2 enters, the conflict data item set is found which contains A. therefore at t=2, the entry in the table for transaction T2(L) is (L, (A), null).
At the same time transaction T1 gets its value for virtual lock modified with the help of function LDeg. So the entry becomes (L, null, null).
The execution follows in the similar manner. At time t=6, when transaction T2 commits a high level secured transaction T3 enters into the system, but transaction T1 gets more priority for execution because the value of virtual level is 'low' for this transaction.
In this example we can see that:

- Covert channel is not established.
- High transactions are not starved.
- No retrieval anomaly exists.

```
T1(H):                                                      R[A1]W[A2]C
           R[A]
T2(L):           R[A]W[A1]                        R[B]W[B1]C
T3(H);                       R[B1]W[B2]R[C]W[C1]C
```

Fig. 1. Execution of Transaction in Multilevel Secure Environment

Table I: Execution Of Transactions Shown In Fig. 1.

Time	T1(H)	T2(L)	T3(H)
1	(H, NULL, NULL)		
2	(L, NULL, NULL)	(L, (A), NULL)	
3	(L, NULL, NULL)	(L,(A),(A1))	
4	(L,NULL,NULL)	(L,(A),(A1))	
5	(L,NULL,NULL)	(L,(A),(A1,B1))	
6	(L,NULL,NULL)	C	
7	R[A1]		
8	W[A1]		
9	C		
10			(H,NULL,NULL)
11			(H,NULL,(B1))
12			(H,NULL,(B1))
13			(H,NULL,(B1,C1))
14			C

B. Number of security levels=5, Number of transactions=5

In this scenario, we have 5 levels for security classifications, with the partial ordering relation as follows:
VERYHIGH (VH) > HIGH (H) >LOW (L) >VERYLOW (VL) > UNCLASSIFIED (U)
The transactions T1, T2, T3, T4 and T5 are issued by VH, H, L, VL AND U users respectively, as shown in fig. 2.
In this example also, we can see that execution follows in the similar manner.
Initially at time t=1 :

VL (Virtual Level) = AL (Actual Level);
c-data-item set = null ;
V-lock = null ;
Therefore at t=1, the entry in the table for transaction T1 (VH) is (VH, null, null).
At time t = 2, when transaction T2 enters, the conflict data item set is found which contains A. therefore at t=2, the entry in the table for transaction T2 (H) is (H, (A), null).Each time a low level transaction enters into the system, all the pending transactions gets their virtual level value degraded by one level by using the function Ldeg.
After time t= 15, as shown in table 3, when the lowest level security transaction gets committed, the values of virtual levels for all the pending transaction are 'UNCLASSIFIED'.
At this time, we can notice two points:

1. No new transaction with security level less than T5 can enter into the system.
2. As all the pending transactions have same value for Virtual Level, so their Actual Level value will be compared and the transaction with highest value of security level will be given the priority. That's why the next transaction to be executed is T1.

Therefore, Covert channel is not established to a definite level, High transactions are not starved and No retrieval anomaly exists.

Interpretation: In this algorithm, the retrieval anomaly is totally eliminated. To make the approach starvation free Ldeg function is used. When the value of virtuallock becomes the unclassified(the lowest security level), it starts executing the corresponding transaction rather than the transaction whose actual lock value is lock. In this way, it may introduce some covert channel. So there may be a tradeoff between the covert channels established and starvation.

As we have already seen in the example that there is a trade-off between covert channel and starvation. Let us consider total no.of security levels be 'n', then an executing transaction will allow maximum 'n-1' more transaction to interrupt it and be executed. Therefore, after the completion of all low level transactions, which is 'n-1' at the maximum, the virtual level of executing transaction will come down to the least security level available. At this moment the covert channel may interfere.

When the covert channels arise in the system, automatically the starvation problem gets eliminated, because the highest level transaction gets the chance to be executed. Although there are low level transactions present in the queue, which are waiting for their execution. As shown in example 2, after t=15, no new transaction will be entertained, due to the virtual level of T1 as the 'very low'. So, starvation is eliminated.

```
T1(VH):   R[A]                                                                                          R[A1]W[A2]
T2(H):
                R[A]W[A1]R[B]                                                               R[B1]W[B2]C
T3(L);
                               R[B]W[B1]R[C]                                  R[C1]W[C2]C
T4(VL):
                                              R[C]W[C1]R[D]          R[D1]W[E2]C
T5(U):
                                                      R[D]W[D1]R[E]W[E1]C
```

Fig. 2. Execution of Transaction in Multilevel Secure Environment.

TABLE II: EXECUTION OF TRANSACTIONS SHOWN IN FIG. 2.

Time	T1(VH)	T2(H)	T3(L)	T4(VL)	T5(U)
1	(VH, NULL, NULL)				
2	(H, NULL, NULL)	(H, (A), NULL)			
3	(H, NULL, NULL)	(H,(A),(A1))			
4	(H, NULL, NULL)	(H,(A),(A1))			
5	(L,NULL,NULL)	(L,(A),(A1))	(L,(B),(A1))		
6	(L,NULL,NULL)	(L,(A),(A1))	(L,(B),(A1,B1))		
7	(L,NULL,NULL)	(L,(A),(A1))	(L,(B),(A1,B1))		
8	(VL,NULL,NULL)	(VL,(A),(A1))	(VL,(B),(A1,B1))	(VL,(C),(A1,B1))	
9	(VL,NULL,NULL)	(VL,(A),(A1))	(VL,(B),(A1,B1))	(VL,(C),(A1,B1,C1))	
10	(VL,NULL,NULL)	(VL,(A),(A1))	(VL,(B),(A1,B1))	(VL,(C),(A1,B1,C1))	
11	(U,NULL,NULL)	(U,(A),(A1))	(U,(B),(A1,B1))	(U,(C),(A1,B1,C1))	(U,(D),(A1,B1,C1))
12	(U,NULL,NULL)	(U,(A),(A1))	(U,(B),(A1,B1))	(U,(C),(A1,B1,C1))	(U,(D),(A1,B1,C1,D1))
13	(U,NULL,NULL)	(U,(A),(A1))	(U,(B),(A1,B1))	(U,(C),(A1,B1,C1))	(U,(D),(A1,B1,C1,D1))
14	(U,NULL,NULL)	(U,(A),(A1))	(U,(B),(A1,B1))	(U,(C),(A1,B1,C1))	(U,(D),(A1,B1,C1,D1,E1))
15	(U,NULL,NULL)	(U,(A),(A1))	(U,(B),(A1,B1))	(U,(C),(A1,B1,C1))	C
*********					
16	R[A1]				
17	W[A2]				
18	C				
19		R[B1]			
20		W[B2]			
21		C			
22			R[C1]		
23			W[C2]		
24			C		
25				R[D1]	
26				W[D2]	
27				C	

*********IF NEW TRANSACTION COMES WITH LEVEL LESS THAN T5, THEN ALSO THAT TRANSACTION WILL NOT EXECUTE . ACCORDING TO THE ALGORITHM TRANSACTION NEXT TO BE EXECUTED IN THAT CASE IS T1, ALTOUGH IT HAS HIGHEST LEVEL OF SECURITY.

VI. CONCLUSION

In this paper we discussed the various traditional approaches for concurrency control and have seen that why they are not applicable to secure databases. We have also proposed a new algorithm for starvation free concurrency control. The basic requirements for security like retrieval anomaly, timing channel are tried to remain intact. In future, the algorithm may be implemented and its performance can be checked.

REFERENCES

[1] Dawyer P., Jelatis G. D. and Thuraisingham B., "Multilevel Security in Database Management Systems", Computer and Security, vol. 6, no. 3, pp. 252-260, June 1987.

[2] S. Jajodia and V. Atluri, "Alternative correctness criteria for concurrent execution of transactions in multilevel secure databases," Proceedings of the IEEE Symposium on Security and Privacy, pp. 839-854,Oakland, California, 1992.

[3] S. Jajodia and C. McCollum, "Using two-phase commit for crash recovery for federated multilevel secure database management systems," Dependable Computing and Fault Tolerant Systems, vol. 8, pp. 365-381, New York, Springer-Verlag, 1993.

[4] J. McDermott and S. Jajodia, "Orange locking: Channel free database concurrency control via locking," Database Security, VI: Status and Prospects, Database Security, pp. 267-284, 1995.

[5] S. Jajodia, L. V. Mancini, and I. Ray, "Secure locking protocol for multilevel database management systems," Proceedings of the Annual IFIP WG 11.3 Conference of Database Security, pp. 177-194, 1995.

[6] S. H. Son and R. David, "Design and analysis of a secure two-phase locking protocol," 18th International Computer Software and Applications Conference(COMPSAC'94), pp. 374-379, IEEE Computer Society Press, 1994.

[7] E. Bertino, B. Catania and E. Ferrari, " A nested transaction model for multilevel secure database management systems," ACM Transactions on Information and System Security, vol. 4, no. 4, pp. 321370, Nov. 2001.

[8] Bertino E. and Sandhu R., "Database Security-Concepts, Approaches, and Challenges", IEEE Transactions on Dependable and Secure Computing, vol. 2, no.1, pp. 2-19, 2005.

[9] S. Jajodia and B. Kogan, "Concurrency control in multilevel secure databases based on a replicated architecture," Proceedings of the llth IEEE Symposium onSecurity and Privacy, pp. 360-368, Oakland, CA, Apr. 1990.

[10] T. F. Keefe and W. T. Tsai, "Multiversion concurrency control for multilevel secure database systems," Proceedings of the IEEE Symposium on Security and 344, Feb. 1990.

[12] T. F. Keefe, W. T. Tsai and J. Srivastava, "Database concurrency control in multilevel secure database management systems," IEEE Transactions on Knowledge and Data Engineering, vol. 5, no. 6, pp. 1039-1055, 1993.

[13] H. T. Kim and M. H. Kim, "Starvation-free secure multi version concurrency control," Information Processing Letters, vol. 65, pp. 247-253 pp. 247-253, 1998.

[14] Imran S. and Hyder I.,"Security Issues in Databases", Proc. of Second International Conference on Future Information Technology and Management Engineering, pp. 541-545, 2009.

[15] Alom B.M.M., Henskens F. and Hannaford M., "Deadlock Detection Views of Distributed Database", Proc. of sixth International Conference on Information Technology: New Generations, pp. 730-737, 2009.

[16] Bertino E. and Sandhu R., "Database Security-Concepts, Approaches, and Challenges", IEEE Transactions on Dependable and Secure Computing, vol. 2, no.1, pp. 2-19, 2005.

[17] Bell D. and La Padula L., "Secure Computer Systems: Unified Exposition and Multics Interpretation", Technical Report NTIS ADA023588. Bedford, Mass.: MITRE Corporation, July 1975.

[18] Kaur N., Saini H. S. and Singh R., "Design And Analysis Of Secure Scheduler For MLS Distributed Database Systems", Proc. Of International Advance Computing Conference, pp. 1400 – 1404, March 2009.

[19] N. Kaur, R. Singh, M. Misra and A. K. Sarje, "Concurrency Control for Multilevel Secure Databases", International Journal of Network Security, Vol.9, No.1, PP.70-81, July 2009.

[20] N. Kaur, R. Singh, M. Misra and A. K. Sarje, "PerformanceEvaluation of Secure Concurrency Control Algorithm forMLS Databases", Proceedings of International Conference on Information Technology, IEEE, 2005.