Wuhan University
Journal of Natural Sciences

2008, Vol.13 No.5, 001-005

Article ID 1007-1202(2008)05-0001-04
DOI 10.1007/s11859-008-0124-2

Formal Analysis on an Extended Security Model for Database Systems

☐ **ZHU Hong, ZHU Yi, LI Chengyang, SHI Jie, FU Ge, WANG Yuanzhen**

Database and Multimedia Technology Research Institute, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China

Abstract: In order to develop highly secure database systems to meet the requirements for class B2, the BLP (Bell-LaPudula) model is extended according to the features of database systems. A method for verifying security model for database systems is proposed. According to this method, analysis by using Coq proof assistant to ensure the correctness and security of the extended model is introduced. Our formal security model has been verified secure. This work demonstrates that our verification method is effective and sufficient.

Key words: Bell-LaPudula model (BLP); security model; Coq; integrity constraints

CLC number: TP 309.2

Received date: 2008-00-00

Foundation item: Supported by the National High Technology Research and Development Program of China (2006AA01Z430)

Biography: ZHU Hong (1965-), female, Professor, Ph.D., research direction: database security. E-mail: zhuhong@public.wh.hb.cn

0 Introduction

In many database applications sensitive information needs to be processed. Database management system (DBMS) are required to provide high level of security. Before designing a highly secure database system to meet the security requirements for class B2[1], a formal security model is needed.

Since the BLP(Bell-LaPudula) model[2] is recommended by the trusted computer system evaluation criteria (TCSEC) [1], many researchers applied it to operating systems and formally analyzed the security of the extended models. Li *et al*[3] specified the BLP in Zlanguage and illustrated that proving by using tools was more rigorous than proving manually. He *et al*[4] analyzed an extended BLP in Zlanguage and verified that their model was not secure. Maximiliano[5] proposed an extended model for a UNIX file system and analyzed that its operations satisfied a set of security properties with Coq[6]. Boniface Hicks specified the SELinux MLS policy in the Prolog and proposed a method to determine policy compliance in different systems[7].

The BLP model has also been applied in secure database systems. In SeaView, the formal verification of the SQL in (EHDM) enhanced hierarchical development methodology is presented[8, 9]. Cheng *et al* extended the object hierarchy in a secure model for a database system and verified the transition rules manually[10]. However, when we apply the BLP model to modern databases, a problem arises. In the BLP model, objects form a tree, but a tree would not describe all the relationships among objects in database systems. Therefore, the objects and their structures of the BLP model need to be extended,

and the descriptions of the system, the security properties and the state transition rules all need to be modified.

Because of the extension, the security of the model should be verified. In this study, a method for verifying the security model for database systems is proposed and the model is verified secure by using Coq.

1 Extended Security Model

Our security model consists of the description of the system, security properties and state transition rules. In the following, the extended and modified points based on the BLP of these three parts are introduced.

1.1 Preliminaries of the Model

A database system is abstracted as a state machine and described as a three-tuple, Sys= (V, ω, D), where D is the set of responses of the system, V the set of all reachable states, ω the set of state transition rules.

The state of a database system is a five-tuple v= (B, M, F, S, O). The B, M and S are the same as the elements in the BLP. In the following, we will illustrate the notations that are not the same as the ones in the BLP.

O is theset of all objects. The types of the objects are divided into two categories: stored objects (e.g., tuples, tables and databases) and derived objects (e.g., views, stored procedures and triggers).

There are two types of structures for objects in the model: tree and graph. All the stored objects form a tree. The database is the root, the tables are the children of the database, and tuples in a table are the children of the table. A derived object is derived from stored objects or other derived objects. The relationships between derived objects and deriving objects form a graph.

After we have analyzed the performance and storage cost, we label the objects at the minimal label granularity of tuple level. If o is a tuple or a derived object, the lowest security level of o (fol (o)) and the highest security level of o (foh (o)) are equal. If o is a table or a database, the security level of o is a range of security levels.

1.2 Security Properties

Because of the extension, the object-compatibility property in the BLP is modified. Since data integrity is the requirement of database system, entity-integrity property and reference-integrity property are added.

1.2.1 Integrity constraints

Object-compatibility property: A state v satisfies the object-compatibility property, if and only if in the state v, object o_2 is the parent of o_1, and one of the conditions below should be satisfied:

① If o_1 is a stored object, then foh(o_2)≥foh(o_1) and fol(o_1)≥fol(o_2);

② If o_1 is a derived object, then fol(o_1)≥fol(o_2).

Entity-integrity property: A state v satisfies the entity-integrity property, if and only if in the state v, for all tuples in any table, the primary key and the security level of the tuple together identify the tuple uniquely. The primary key should not be empty.

Reference-integrity property: A state v satisfies the reference-integrity property, if and only if in state v, if o_1, o_2 exist and are tuples, the table containing o_1 is referenced by the table containing o_2,.The primary key of o_1 equals to the foreign key of o_2 and fol(o_1)=fol(o_2).

Only one foreign key is denoted, since the notations of the property for several foreign keys are similar.

1.2.2 Security axioms

Simple-security property: The simple-security property in the model is the same as the one in the BLP.

Star-property: State v satisfies the star-property, if and only if for the state v and any subject s, one of the conditions below should be satisfied.

① If s accesses o in mode r, then $f_c(s)$≥ fol(o);

② if s accesses o in mode a, then foh(o)≥$f_c(s)$≥ fol(o);

③ if s accesses o in mode w, then $f_c(s)$=fol(o);

④ if s accesses o in mode c or e, then v satisfies the star-property.

Discretionary property: State v satisfies the discretionary property, if and only if for the state v and any subject s accessing an object o in mode x, either of the conditions below should be satisfied:

① if o is a tuple, then s has the privilege to access the table which contains o in mode x;

② if o is not a tuple, then s has the privilege to access o in mode x.

1.3 State Transition Rules

We defined nine state transition rules according to the basic operations in databases, including: query, insert, update, alter, execute, delete, drop, grant and revoke. We take query as an example due to the space limit.

Query rule: Under v_i, an s_i can query a table or a view o_j, when the following conditions are satisfied:

① o_j is a table or a view existing in the database;

② s_i has the privilege to access o_j in mode r;

③ $f_c(s_i)$≥ fol(o_j).

If the response is "yes", v_i is changed into v_j.

When o_j is a table, for any tuple o'_j whose parent is o_j, $f_c(s_i)$≥fol(o'_j), (s, o'_j, r) should be added into set B in

v_i to obtain a new B in v_j. The other four elements of v_j are the same as the ones in v_i.

When o_j is a view, for any tuple o'_j whose parent has derived o_j, $f_c(s_i) \geqslant \text{fol}(o'_j)$, (s, o'_j, r) should be added into set B in v_i to obtain a new B in v_j. The other four elements of v_j are the same as the ones in v_i.

If any condition in ①, ② and ③ above is not satisfied, the decision would be "no" and the state is still v_i.

2 The Extended Theorems for Security

In our model, no information in high security level would be transferred to the subjects with low security level. Therefore, the security invariants for verification would partly be insured by three security axioms in our model.

The difference between database systems and other systems is the requirements for data integrity. The ignorance of the entity and reference integrity constraint would cause covert channels. Therefore, the security invariants should be extended to include integrity constraints. We should verify the integrity of the state first.

Theorem 1 A state v_i satisfies the integrity constraints if and only if the state satisfies the object-compatibility property, the entity-integrity property and the reference-integrity property.

Theorem 2 A state v_i is secure if and only if the state satisfies the integrity constraints, the simple-security property, the star property and the discretionary property.

Theorem 3 A state v_i is a reachable state if and only if it is the initial state v_0 or it is transferred from the initial state v_0 by a sequence of state transition rules ρ_{i1}, ρ_{i2},…, ρ_{in}.

Theorem 4 A state transition rule ρ_i is secure if for any secure state v_i, a state v_{i+1}, which is changed from v_i under the state transition rule ρ_i, is also secure.

Theorem 5 Suppose state v_{i+1} is changed from v_i under ρ_i, and ρ_i is secure,

① If v_i satisfies the integrity constraints, then v_{i+1} satisfies the integrity constraints,

② If v_i satisfies the simple-security property and v_{i+1} satisfies the integrity constraints, then v_{i+1} satisfies the simple-security property,

③ If v_i satisfies the star-property and v_{i+1} satisfies the integrity constraints, then v_{i+1} satisfies star-property.

④ If v_i satisfies the discretionary property and v_{i+1} satisfies the integrity constraints, then v_{i+1} satisfies the discretionary property.

Inference 1 A sequence of state transition rules is secure if and only if every rule in the sequence is secure.

Theorem 6 A system Sys is secure if and only if all the reachable states in the system are secure.

3 Formal Analysis of the Extended Security Model

3.1 Formal Verification

We have verified that the extended model is secure based on the theorems in Section 2.

The initial state v_0 should be verified secure first. In our model, there are only untrusted subjects in the system. Both the set B and S are empty in v_0. Therefore, that v_0 satisfies the integrity constraints and the security axioms should be verified. Then, that other states are secure should be verified. We take the query rule as an example to illustrate analysis of the model due to the space limit.

Firstly, the rule query is defined as a predicate with two parameters in Coq: subject and object (shown in Fig.1) according to the definition in Section 1.3.

In Fig.1, v_i. represents thestate of v_i for example, $o \in v_i$.Tables describes object o is a table in the set of tables of state v_i. Parentof(t) returns the parent of t. DerivedSO(o) returns the set of the stored objects that derives o. Parento*f*(t) ∈ DerivedSO(o) describes the parent of t is anyone of the stored objects that the view is based on. (AddB v_i (s, t, r)) describes the relationship between v_j and v_i, which indicates that (s, t, r) is added into set B of v_i to form v_j. If the proposition (query s o) is true, s queries o and v_i is changed into v_j.

```
Definition query (s : subject_element) (o: object_element) :Prop
  := ((o ∈ v_i.Tables /\ o ∈ v_i.O /\ (s, o, r) ∈ v_i.M /\ v_i.f_c(s)≥v_i.fol(o)
        /\ (forall t : object_element,
      (t ∈ v_i. Tuples /\ o=Parentof (t) /\ v_i. f_c(s)≥v_i.fol(t)
          /\ v_j=AddB v_i (s, t, r)))
\/ ((o ∈ v_i.Views /\ o ∈ v_i.O /\ (s, o, r) ∈ v_i.M) /\ v_i. f_c (s)≥v_i. fol (o)
      /\(forall t: object_element,
        (t ∈ v_i. Tuple /\ (Parentof(t)) ∈ DerivedSO(o)/\ v_i. f_c(s)
≥v_i.fol(t)) /\( v_j=AddB v_i (s, t, r))).
```

Fig.1 The definition of query rule in Coq

The query rule in Fig.1 is defined according to different types of the extended objects. The returned results for the query rule are actually Tuples which should be added into set B. The parent-and-son relationships are

different in different structures because of the extension of the object structure. Tuples can not be the son of derived objects. As a result we first find all the Tables that derive the view, and then find the Tuples in the Tables to get the queried Tuples.

Secondly, we should verify that v_j transiting from v_i through query rule is secure if v_i is secure. According to Theorem 1 and Theorem 2, in order to prove v_j is secure, we should verify that v_j satisfies the integrity constraints first and continually prove that v_j satisfies the simple-security property, star-property and discretionary property respectively. We take the verification of discretionary property in Fig.2 as an example.

Lemma Query_Discretionary:
forall (*s* : subject_element) (*o* : object_element),
Secure v_i /\ Integrity_State v_j
/\Query *s o* → Discretionary v_j.

Fig.2 Lemma query_discretionary in Coq

In Fig.2, Query_Discretionary is the defined name of a proposition to be proved with keyword Lemma. (Discretionary *v*) indicates that v_i satisfied the discretionary property. (Secure v_i) indicates that v_i is secure. (Integrity_State v_j) indicates that v_j satisfied the integrity constraints. (query *s o*) indicates that *s* queries *o* successfully. Lemma Query_Discrestionary is to prove (Discretionary *v*) under the three conditions. Coq provides tactics to reduce the current goal into sub goals. We prove (Discretionary *v*) with some reducing and applying tactics. Because the Coq tool operates on the current goal by attempting to construct a proof of the current goal from corresponding conditions, if one condition was missing, the current goal would not be proved. This proving logic makes our verification sure stricter and more precise.

The verification that v_j satisfies simple-security property and star property are almost the same.

Finally, v_j is secure, and the query rule is secure according to Theorem 4. Other eight state transition rules could be also verified secure analogously. According to Inference 1, any sequence of the state transition rules Seq_i is secure. If any v_i is secure, a reachable state v_j transferred by Seq_i is secure according to Theorem 5 and Theorem 2. Since v_0 is secure, if any v_i is secure, state v_j transferred from v_i by Seq_i is also secure, all the states are secure. Finally, the system is secure.

3.2 Results of Formal Verification

Both the specification and verification for the model were developed in Coq. We spent two months on the proof with 72 lemmas and 2 100 lines Coq code. The whole verification is executed in the computer with Intel Core2 Duo processor (1.86 GHz), 2 GB memory and Microsoft Windows XP operating system within 20 min.

4 Conclusion

An extended BLP model is presented and a method for verifying a security model for database systems is proposed. According to the method, the model has been verified secure by using Coq. This work demonstrates that our verification method is effective and sufficient.

In the future, we plan to verify that all the SQL in database systems are accordant with our security model.

References

[1] Department of Defense of USA. Trusted Computer System Evaluation Criteria[EB/OL].[2007-01-12].*http://en.wikipedia.org/wiki/Trusted_Computer_System_Evaluation_Criteria.*

[2] Bell D E, LaPadula L J. Secure Computer Systems : A Mathematical Model[EB/OL]. [2007-02-12]. *http://www.albany.edu/acc/courses/ia/classics/belllapadula2.pdf.*

[3] Li Liping, Qing Sihan, Zhou Yi. Research on Formal Security Policy Model Specification and Its Formal Analysis[J]. *Journal on Communication*, 2006, **27**(6): 94-101(Ch).

[4] He JianBo, Qing Sihan, Wang Chao. Formal Safety Analysis of a Class of Multilevel Security Models[J]. *Chinese Journal of Computers*, 2006, **29**(8): 1468-1479(Ch).

[5] Maximiliano C. Verification of an Extension of a Secure, Compatible UNIX File System. [EB/OL]. [2007-10-20]. *http://www.fceia.unr.edu.ar/gidis.*

[6] Huet G, Kahn G, Paulin-Mohring C. The Coq Proof Assistant[EB/OL].[2007-05-01]. *http://coq.inria.fr/.*

[7] Boniface H, Sandra R, Luke S C, *et al*. A Logical Specification and Analysis for SELinux MLS Policy[EB/OL]. [2007-10-20]. *http://www.patrickmcdaniel. Org/pubs/sacmato*7*.pdf.*.

[8] Lunt T F, Denning D E, Schell R R, *et al*. The SeaView Security Model[J]. *IEEE Transactions on Software Engineering*, 1990, **16**(6): 593-607

[9] Whitehurst R A, Lunt T F. The SeaView Verification [EB/OL]. [2006-11-20]. *http://ieeexplore.ieee.org/iel*2/869/1554/00040595*.pdf.*

[10] Cheng Wanjun, Zhang Xia, Liu Jiren. A Secure Policy Model for Secure Database System Based on Extended Object Hierarchy[J]. *Journal of Software*, 2003, **14**(5): 955-962(Ch).

□