Using Petri Nets to Verify Access Policies in Mandatory Access Control Model

Yixin Jiang, Chuang Lin, Zhen Chen, Hao Yin
Institute of Computer Network, Tsinghua University, Beijing, P. R. China, 100084
{yxjiang, chlin, zhenchen, hyin}@csnet1.cs.tsinghua.edu.cn

Abstract

MAC (Mandatory Access Control) model is a basic security model for describing and verifying information system. In this paper a method is presented to verify security policies in MAC Model by means of Coloured Petri Nets (CPN). The main theme of this paper is depicted as follows: Firstly, based on the lattice model of multi-level security (MLS) and Bell-LaPadula model, the MAC model is formally defined. Subsequently, an equivalent MAC model described by Colored Petri Nets (CPN) is proposed. According to the derived state reachability graph, four security properties of access control policy relations in MAC model are investigated, i.e. access temporal relations, subject reachability analysis, covet channel analysis, inference of sensitive information. An example of the security model is given for illustration. The results show that this concise graphic analysis method is suitable for formal verification. This method can be efficiently used to analyze information flow security and therefore improve the whole access control policies during security design and implementation of the system.

1. Introduction

Security model, as a measure to guarantee the privacy and integrity of data, is a formal method to describe and verify a complex information system. It is widely used not only in the abstract definition of system security requirements, but also in design and implementation of information system.

Traditionally, two different security models can be identified depending on who is in charge of setting security rules, i.e. mandatory access control (MAC) model and discretionary access control (DAC) model. In these two security models, all system components can be divided into two categories: objects and subjects. In DAC model, the owner of objects freely decides who has right to access them, while in MAC model is intended to guarantee secure information flow within the system. An important concept relative to MAC model is Multi-Level Security (MLS), i.e. each object and subject is assigned a security level, whereby the security classes form a partially ordered set (or lattice). By using MLS, MAC model allows the transfer of information only in one direction in the lattice of security classes.

According to the well-known Trusted Computer System Evaluation Criteria (TCSEC) [1], in most cases we need a formal model to verify the correctness of system security policy. That is, a model can describe initial state of the system, security states, and transitions from one state to another. Formal methods are often used to conduct automated analysis. So far a number of techniques to support the security analysis of MAC model were proposed, such as Varadharajan method [2] or a predicate-transition net method [3]. However, applying these techniques to model complex MAC-based information systems is not a trivial task. Usually, we must take into account the existence of hundreds of system components (subjects or objects) and the need of frequently changing system security policies. Several problems emerging when trying to implement MAC rules in distributed environments were discussed in [4, 5]. Effective management mechanisms must be implemented while preserving mathematical soundness of the model [6, 7]. Such factors dramatically affect the complexity of existing models and make them hardly applicable. Hence, we need a formal approach to:

- Conducting automated information flow analysis.
- Using specialized software tools to support constructing and managing security models for complex systems (with reduced complexity).
- Guaranteeing the semantics equal to the MAC models.

In order to fulfill these requirements, a new security analysis method based on Colored Petri Nets (CPN) is proposed in this paper. Moreover, according to the derived state reachability graph with semantics equal to MAC model, we chiefly explore four important security properties of access control policy relations, i.e. the access temporal relations, reachability analysis when subject accesses objects, hidden security holes due to the change of security level, the indirect inference of confidential information between different objects. In [8], the author only proposed a CPN-based model to describe MAC model, but in this paper, we step further and apply it to analyze the security properties in detail. Therefore, by using our method, many security drawbacks can be
eliminated in advance and the whole access control policies can also be improved during the security design and implementation of the system.

2. Mandatory Access Control Model

MAC model includes two entities: objects and subjects. **Objects** mainly include passive entities (file, storage area) while **Subjects** mainly contain active entities (processes, users). Subjects obtain information by accessing objects. MAC model has an evident peculiarity that each subject or object is assigned a security level. System security administrator directly impacts on a subject’s access mode by comparing the security level of a subject with that of an object. Unlike DAC model, subjects have no privilege to decide the right when accessing objects.

2.1. Multi-Level Security

Multi-Level Security (MLS) has a long tradition in military environments and is a basic requirement for A and B security classes in the TCSEC [1]. MAC model is based on security class, which is involved in two concepts, i.e. security domains and security levels. For convenience, we firstly introduce the concept of partially order set and lattice respectively.

**Def. 2.1** The pair \((R, \leq)\) is a partially order set if “\(\leq\)” defines a reflective, anti-symmetric, and transitive relations on set \(R\). A partial order set \((R, \leq)\) is called **lattice** if for every \(x, y \in R\) there exists a least upper bound \(l \in R\) and a greatest lower \(g \in R\).

Security domain denotes the legal active range of entities. If \(D_i\) denote a finite set of all domains and \(D=\{D_i | D_i \in power(D_s)\}\) (power denotes the power set), we can define security domain as follows.

**Def. 2.2** The **security domain** is a lattice defined by five-tuple \((D_i, \otimes, \oslash, \Phi, D_o)\). For \(\forall D_i, D_j \in D\), the operator \(\otimes, \oslash\) and relation “\(\leq\)” could be defined as follows:

\[
D_i \otimes D_j = D_i \cup D_j; D_i \oslash D_j = D_i \cap D_j; D_i \leq D_j \rightarrow D_i \subseteq D_j.
\]

**Def. 2.3** The **security class** is a lattice defined by the five-tuple \((L, \oplus, \ominus, \Phi, low, high)\). Let \(S=\{S_i | i=1,2,\ldots,n\}\) denote the set of the security levels. \(low = S_1\) represents the lowest security level; \(high = S_n\) represents the highest security level. For \(\forall S_i, S_j \in S\), the operator \(\oplus, \ominus\) and partial order relation “\(<\)” could be defined as follows:

\[
S_i \oplus S_j = S_{\text{max}(i,j)}; S_i \ominus S_j = S_{\text{min}(i,j)}; S_i < S_j \rightarrow i < j.
\]

**Def. 2.4** The **MLS class** is a lattice defined by the five-tuple \((L, \oplus, \ominus, \Phi, low, high)\). \(L\) is the product of \(S\) and \(D\), i.e. \(L=S \times D=\{(S_i, D_j) | S_i \in S, D_j \in D\}\). For \(\forall (S_i, D_j), (S_j, D_j) \in L\), the operator \(\oplus, \ominus\) and partial order relation “\(<\)” could be defined as follows:

\[
(S_i, D_j) \oplus (S_j, D_j) = (S_{\text{max}(i,j)}, D_i \cup D_j).
\]

The element of security class is a two-dimension vector \((S, D)\), which consists of a security level and a security domain. In MAC model, the security levels usually contain four members, i.e. Unclassified (U), Confidential (C), Secret (S), TopSecret (TS), with \(U < C < S < TS\). The security level is assigned to each subject or object.

2.2. MAC model

According to the lattice model of MLS and Bell-LaPadula model [9], we introduce the formal definition of MAC model. The two access rules are **No-read-up** and **No-write-down** which demands that low-level subjects are not allowed to read high-level objects, and high-level objects are only written by low-level subjects. These rules keep information flow from low to high.

**Def. 2.5** The **Mandatory Access Control** model is a six-tuple \(M=(S, O, A, L, f, R)\), such that:

- \(S\) is the set of subjects.
- \(O\) is the set of objects.
- \(A\) is access mode set when subjects access objects: \(S \times O \rightarrow \{\phi, \{\text{read}\}, \{\text{write}\}, \{\text{read, write}\}\}\).
- \(L\) is security level, where \((L, \leq)\) is a lattice.
- \(f\) is security functions which map each subject and object to a security level \(l \in L\).
- \(R\) is a set of security rules to prescribe the constraints conditions when subjects access objects. They must obey the two contraints: No-read-up and No-write-down.

**Def. 2.6** **No-read-up** rule denotes that a subject \(s \in S\) is granted read-access to object \(o \in O\) if \(f(o) \leq f(s)\). **No-write-down** rule denotes that a subject \(s \in S\) is granted write-access to object \(o \in O\) if \(f(s) < f(o)\).

The above rules are very robust but have several flaws. The main drawback is covert channels. A covert channel represents a leakage to the defined information flow and cannot be controlled by security policies. In later sections, we would discuss this problem in detail.

**Def. 2.7** In MAC model, the security level of a subject is a range (minL, maxL). If minL<maxL, the subject is called **Trusted Subject**. If minL=maxL, the subject is called **Untrusted Subject**.

3. Colored Petri Nets

Petri Net, as a formal tool, has a well-defined rigorous semantics and can be efficiently used to model and verify the security properties of a system model. Colored Petri Nets (CPN) extends Petri Nets by allowing tokens to be associated with colors, i.e., data types using a functional programming language, SML. Additionally, transitions
and arcs can be augmented with guards and expressions, respectively [10].

**Def. 3.1** A CPN is a nine-tuple \((\Sigma, P, T, A, N, C, G, E, I)\) satisfying these requirements:
- \(\Sigma\) is a finite set of non-empty types, called color sets.
- \(P\) is a finite set of places.
- \(T\) is a finite set of transitions.
- \(A\) is a finite set of arcs: \(P \cap T = P \cap A = T \cap A = \emptyset\).
- \(N\) is a node function: \(A \rightarrow (P \times T) \cup (T \times P)\).
- \(C\) is a color function: \(p \rightarrow \Sigma\).
- \(G\) is a guard function. It is defined from transition set \(T\) into expressions, such that: \(\forall t \in T, Type(G(t)) = \text{Boolean} \land Type((\text{Var}(G(t)))) \subseteq \Sigma\).
- \(E\) is an arc expression function. It is defined from arc set \(A\) into expressions, such that: \(\forall a \in A, Type(E(a)) = C(p(a))_{\text{as}} \land Type((\text{Var}(E(a)))) \subseteq \Sigma\), where \(p(a)\) is the place of \(N(a)\).
- \(I\) is an initialization function. It is defined from \(P\) into closed expressions, such that: \(\forall p \in P, Type(I(p)) = C(p)_{\text{as}}\).

**Def. 3.2** A binding of a transition \(t\) is a function \(b\) defined on \(\text{Var}(t)\), such that:
\[
\forall v \in \text{Var}(t): b(v) \in Type(v) \land G(t) \land Type(b(v)) \subseteq \Sigma.
\]
where \(expr\) denotes the evaluation of the expression \(expr\) in the binding \(b\). Let \(B(t)\) mean the set of all bindings for transition \(t\).

**Def. 3.3** A token element is a pair \((p, c)\) where \(p \in P\) and \(c \in C\), while a binding elements is a pair \((t, b)\), where \(t \in T\) and \(b \in B(t)\). The set of all token elements is denoted by \(TE\) while the set of all binding elements is denoted by \(BE\). A Marking is a multi-set over \(TE\) while a step is a finite multi-set over \(BE\).

**Def. 3.4** A step \(Y\) is enabled in a marking \(M\) iff:
\[
\forall p \in P, \forall t \in T, \Sigma(Y, t) \cdot E(p, t) \cdot b > 0 \Leftrightarrow M(p).
\]

when a binding element \((t, b)\) is enabled in a marking \(M_1\), it may fire and change the marking \(M_1\) into \(M_2\) defined by:
\[
\forall p \in P, M_2(p) = (M_1(p) - \Sigma(Y, t) \cdot E(p, t) \cdot b) + \Sigma(Y, t) \cdot E(t, p) \cdot b > 0.
\]

We say that \(M_2\) is directly reachable from \(M_1\) by the occurrence of \((t, b)\), which is denoted by \(M_1[Y > M_2]\).

**Def. 3.5** A finite occurrence sequence is a sequence of markings and steps:
\[
\sigma = M_1[Y_1 > M_{i_1}] \cdots Y_n > M_{i_n} + 1
\]
such that \(n \in N\) and \(M_i > M_{i+1}\) for all \(i \in 1 \ldots n\). A marking \(M'\) is reachable from a marking \(M''\) iff there exists a finite occurrence sequence, e.g., iff for some \(n \in N\) there exists a sequence of steps \(Y_1, Y_2, \ldots, Y_n\) such that: \(M'[Y_1, Y_2, \ldots, Y_n] \geq M''\).
The set of markings which are reachable from \(M'\) is denoted by \([M']\), and \(M'[Y_1, Y_2, \ldots, Y_n] \geq M''\) could be compactly denoted by \([M'][* > M'']\).

**Def. 3.6** The reachability graph of CPN is the directed graph \(OG = (V, A, N)\) where:
\[
\begin{align*}
&V = [M_0^*]. \\
&A = \{(M_1, b, M_2) \in V \times BE \times V \land (M_1[Y > M_2])\}. \\
&\forall a = (M_1, b, M_2) \in A, N(a) = (M_1, M_2).
\end{align*}
\]

## 4. Using CPNs to Describe MAC model

### 4.1. Entity security model

For describing the MAC model with CPN, we firstly introduce the concept of **Entity Security Model (ESM)**. Similar to the concept of Entity Model in Database, the ESM is close to the realistic MAC model while MAC model described by CPNs is more abstract. As shown in Fig.1, the ESM is the intermediate output to convert the MAC model into CPN model with equivalent semantics.

![Fig. 1. Entity security model](image1)

**Def. 4.1** Suppose \(O\) be an object set in MAC, an ESM model is four-tuple \((E, R, f, L)\), such that:
\[
\begin{align*}
&E \subseteq O \text{ is the entity sets of security model}.
\end{align*}
\]

\[
\begin{align*}
&R \subseteq O \text{ is the subordinate relations between entities}.
\end{align*}
\]

\[
\begin{align*}
&f \text{ and } L \text{ is the same as Def. 2.5}.
\end{align*}
\]

![Fig. 2. Entity Security Model](image2)

As an example, an ESM is shown in Fig.2. The elements of this ESM can be listed as follows:
\[
\begin{align*}
&E = \{\text{institute, Prof.1, Prof.2, project, Symposium}\}; \\
&R = \{\text{member, research, attend, subject, director}\}; \\
&L = \{U, C, S, TS\}, with U < C < S < TS; \\
&f(\text{institute}) = U, f(\text{prof.1}) = U, etc.
\end{align*}
\]

The security class is assigned to every object by system administrator. They obey the relations \(U < C < S < TS\), and form a lattice.

### 4.2. Equivalent CPNs model of MAC model

The **semantic-equivalent transition algorithm** from ESM to CPN model is characterized as follows:
\[
\begin{align*}
&\text{Places } P \text{ denote the } E \text{ elements of } ESM.
\end{align*}
\]
Transitions $T$ denote the $R$ elements of $ESM$.  
Color sets $\Sigma = \{ L, Access\_mode, PR \}$, such that: $L$ is security class; $Access\_mode$ is access mode when subjects access objects; $PR$ is the product of $L \times L \times Access\_mode$. Variables $maxL$ and $curL$ denote the maximal and current security class of subjects respectively; $objL$ denotes the security class of objects.

- **Color function** $C(p) = PR = L \times L \times Access\_mode$.
- **Guard function** $G$ denotes the conditions of a transition firing, i.e., the constraint denotes whether subjects have the right to access the $R$ in $ESM$. It obeys the rule listed in Def. 2.6, therefore Guard function can be expressed as $\{maxL \geq relationL\}$, where $relationL$ is the security class of $R$ in $ESM$.
- **Arc functions** $E$ modify the token variable $access$. In outgoing arcs, the expression moves the token $PR$ within the place. In ingoing arcs, the expression enforces MAC security policy and makes the $access$ mode according to the security class of subject and object at the end of the arc.

Based on ESM definition and the semantics of MAC model, we could use above transition algorithm to derive the equivalent CPN model in Fig. 3.

Evidently, the semantics of CPN model derived from the ESM model is equivalent to the original MAC model described by ESM, since it conforms to the two rules defined in Def. 2.6. The incoming arc function uniformly describes the security policy when subjects access objects in spite of trusted or untrusted subjects. The outgoing arc function only moves the token from that place.

### 4.3. Reachability graph of MAC model

After modeling the system with CPN, it is possible to use existing analysis techniques to verify the security properties. Such tool is available, e.g. Design/CPN[10].

![Fig. 3. Equivalent CPN model for MAC model](image)

![Fig. 3. Equivalent CPN model for MAC model](image)

Supposing the subjects of $(institute, Prof.1, Prof.2, project, Symp.)$ in Fig.3 is denoted by the place $(P_1, P_2, P_3, P_4, P_5)$. We can simply define the Marking as follows.

A marking of a Petri Nets is a function $M: P \rightarrow \{0,1\}$, i.e. $\forall p \in P$, $p \rightarrow \{0,1\}$.

- $m_i = M(p_i) = 1$, if the place $p_i$ contains one or more tokens with $access = \{read\}$, $\{write\}$, $\{read, write\}$.
- $m_i = M(p_i) = 0$, if the place $p_i$ contains no token or $access = \phi$.

Let us consider the CPN model shown in Fig. 3. As an example, the reachability graph for a trusted subject $(curL < maxL)$ with security class $[U, TS]$ is shown in Fig. 4.
5. Security analysis of MAC model

5.1. Inference of sensitive information

In Fig. 6, though a subject $S$ is authorized to access the relations $R2$, $R3$, and $R4$ except for $R1$, the subject could infer about the existence of sensitive relation $R1$ by another access path: $E1 \rightarrow E3 \rightarrow E4 \rightarrow E5$.

![Fig. 6. Reasoning of confidential information](image)

As shown in Fig.2, suppose that Prof.1 attends a Symposium to discuss a project, he may infer the relative project of this institute, even though the Prof.1 is not authorized to access the relation research. By analyzing the reachability graph of an un-trusted subject in Fig.5, we can draw a conclusion that the un-trusted subject could infer the proceeding projects of the institute by an indirect path. This hidden security hole could be found by checking whether there exists a path including marking $M_i$ and $M_j$ in the reachability graph, and satisfies:

$$(M_i(P_i)=1) \land (M_i(P_2)=0) \land (M_j(P_1)=0) \land (M_j(P_2)=1)$$

where $P_1$ and $P_4$ denote the entities institute and project respectively.

![Fig. 7. Reachability graph for modified security class](image)

As a result, to refrain the sensitive information from being inferred between the objects institute and project, the following condition, $\forall M_i, M_j$, must be satisfy:

$$(M_i(P_1)=1) \land (M_i(P_2)=0) \land (M_j(P_1)=0) \land (M_j(P_2)=1)$$

$$\rightarrow (((M_j[Y>M_i][Y_i>M_i \ldots > M_j] \land (M_i \neq M_{i+1})))$$

A feasible method to prevent this security problem is to change security class of entity relations. If we change the security class of relation subject from $S$ to $TS$, the subject with security class $S$ could not access the object project, as shown in Fig.7.

5.2. Access temporal relations

According to the reachability graph, it is convenient for us to analyze temporal relations when subjects access objects. Considering the following cases:

- **Follow** relation: $Follow(P_1, P_2) = \{(P_1, P_2) \mid (M_i(P_1)=1) \land (M_i(P_2)=1) \land (M_j[Y>M_i])\}$. This relation denotes that the subject immediately accesses object $P_1$ after accessing object $P_2$.
- **Precede** relation: $Precede(P_1, P_2) = Follow(P_2, P_1)$. The relation is opposite to **Follow** relation.
- **Adjacent** relation: $Adjacent(P_1, P_2) = \{(P_1, P_2) \mid (M_i(P_i)=1) \land (M_i(P_2)=1) \land (((M_j[Y>M_i] \lor (M_j[Y>M_i]))\}$. This relation denotes that the object $P_1$ and $P_2$ being accessed by a subject must be adjacent.
- **After** relation: $After(P_1, P_2)=\{(P_1, P_2) \mid (M_i(P_1)=1) \land (M_i(P_2)=1) \land ((M_i[>Y>M_i] \lor (M_i \neq M_{i+1})\}$. This relation denotes that the subject accesses object $P_1$ after accessing object $P_2$, moreover object $P_1$ and $P_2$ must not be adjacent.
- **Before** relation: $Before(P_1, P_2) = After(P_2, P_1)$. This relation is opposite to **After** relation.
- **Mutex** relation: $Mutex(P_1, P_2)=\{(P_1, P_2) \mid (M_i(P_i)=1) \land \neg After(P_1, P_2) \land \neg Precede(P_1, P_2) \land \neg Before(P_1, P_2)\}$. This relation denotes that the subject do not allow to access object $P_2$ after accessing object $P_1$.

All these temporal relations can be formally verified by using the model-checking method [11, 12].

5.3. Access reachability

**Def. 5.1**: If a subject is authorized to access an object by system administrator and there surely exists a path by which the subject could access the object in reachability graph, we argue that the object with regard to the subject is **access reachability**, otherwise **access unreachability** in spite of the legal right which is authorized by the security administrator.

If the place in CPN denotes the object is $P_i$, an object is **access reachability** in subject’s reachability graph, iff:

$$\exists M_i \in [M_0], (M_i(P_i)=1) \land (M_i[>Y>M_i])$$

In Fig.7, when the security class of $subject$ is altered from $S$ to $TS$, this may result in that a subject with security class $S$ could not access the object $Project$, though the subject is authorized to access the object $Project$ by system administrator. In other words, as far as the subject is concerned, the object is access unreachable.

5.4. Covet channel analysis

Considering the following case: the maximal security class of subject $S$ is $L_0$, while the security class of objects of $O_1$ and $O_2$ are $L_1$ and $L_2$ respectively ($L_0>L_1>L_2$). At moment $t_1$, when the current security class of the subject $S$ is $L_0$, $S$ could read information $I$ from object $O_1$ ($L_0>L_1$).
At moment $t_2$, when the current security class of the subject $S$ changes to $L_2$, in this case, $S$ could write information $I$ to object $O_2 (cur=L_2)$. Therefore, the information $I$ is flowed into the object $O_2$ with low security class $L_1$ from the object $O_1$ with high security class $L_2$. See Fig.8.

Fig.8. Security hole in dynamic security class

In Fig 4, if a trusted subject accesses an object Prof.2 with security class $U$ after accessing the object Project with security class $S$, thus the information of object Project may be flowed into the object Prof.2. It is illegal because of security class $U<S$.

Therefore, if the security class of subjects is dynamic, to prevent the above security hole, we can introduce the constraint condition:

$$\forall p, q \in P, M_i[l_{p+1}, (M_i(p) = 1) \land (M_i(q) = 1) \rightarrow f(p) \leq f(q)$$

where $f(p)$ and $f(q)$ denote the security class of objects $p$ and $q$ respectively.

6. Conclusions and future work

In this paper, based on the lattice model of multilevel security and Bell-LaPadula security model, the MAC model is formally defined. Subsequently, the equivalent MAC model described by CPNs is proposed. According to the reachability graph, four security attributes, i.e. the temporal relations and object's access reachability in the accessing process of subjects, covert channel problem due to the dynamic security level, the reasoning of sensitive information between objects, are investigated in detail. By using this formally analysis method, we can efficiently improve the overall security access policies during the system design and implementation.

The future work would mainly be focused on the following issue. In a small scale MAC model, the state-space issues are not much of a concern. However, when describing a larger MAC model, the size of the state-place will become a big problem. This is an important and difficult issue to be resolved.

7. Acknowledgement

The relative work is supported by the NSFC under contracts No.90104002 and No.60173012 (China); the Projects of Development Plan of the State High Technology Research under contract No. 2003CB314804; Intel University Research Plan (China Research Center).

References


