On the Chinese Wall Model

Volker Kessler
Siemens AG
Dept. ZFE ST SN 5
Otto-Hahn-Ring 6
D-8000 Munich 83

Abstract

We present a modified version of the Chinese Wall model. Especially, we make some investigations on the indirect information flow induced by the write access. In the original Brewer-Nash model the write permission is very restricted. There a subject can get write access to one object only and only during early states of the system. We change this rule and introduce a dynamic concept of the "conflict of interest relation". Thus, we prevent an indirect information flow by building more Chinese Walls. Finally, we prove that the system is "conflict secure", i.e. a subject can never get sensitive information from two or more objects which are in conflict of interest to each other.

1 Introduction

The Chinese Wall model is a well-known security model which combines discretionary and mandatory aspects of access control. In 1989 it was presented by Brewer and Nash [2] who derived it from the British law for stock brokers consulting different companies.

We consider a set of companies partly competing with each other and a group of consultants. It is forbidden that a consultant works for a company if he has insider knowledge of a competitor. Thus the goal of the security policy is to prevent information flows from competing companies to the same consultant.

The consultants are supposed to be users of a computer system storing data of the companies. In the beginning every user has free choice to read any data. After he has read the data of one company a Chinese Wall is built between this chosen company and its competitors and from now on the user is not allowed to read data behind the wall. But he can still read data from companies not competing with the first one, e.g. if he has chosen a bank first, he still has free choice to advise any oil company.

Thus an irregular *direct* information flow is easily prevented. But if the user writes data from the bank B1 to the oil company Z1 an irregular *indirect* information flow can occur because a second consultant might read the data of the oil company Z1 with the data of bank B1 and the data of the competing bank B2 as well, see figure 1. We will come back to this problem later on.

One interesting feature of the Chinese Wall security policy proved by Brewer and Nash is that it cannot be modelled by Bell-LaPadula.

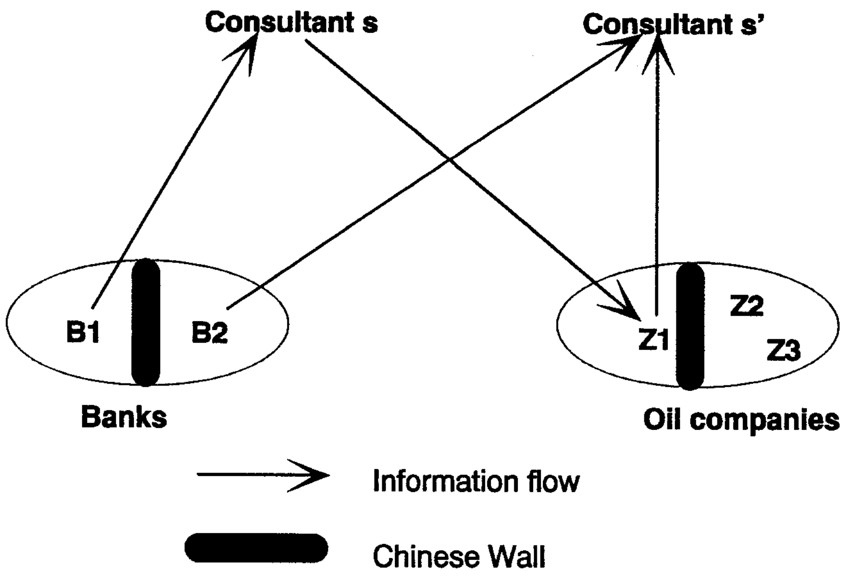

Figure 1: An indirect information flow

Later in 1989 Lin [3] gave an improvement and a more precise formulation for the Chinese Wall model. Further work was done by Catherine Meadows in 1990 by applying the Chinese Wall model in order to handle the aggregation problem in a multilevel context.[1]

We take Lin's version as a basis for our modifications which concentrate mainly on the write permission and the resulting information flows. We use a formal notation similar to Bell-LaPadula.

2 Description of the formal model

2.1 Entities

There is a set S of *subjects* and a set O of *objects*. A *subject* $s \in S$ is an user or a process acting on the behalf of the user s. If the pair (user, process) were regarded as one subject each user could get access to data behind his Chinese Wall by simply starting a new process. Thus in the Chinese Wall model the subject are assumed to have a *memory*. In so far it is different from the Bell-LaPadula approach [1]. The *security manager* is denoted by s_0.[2]

An *object* o is a data file. We can imagine an object as a collection of all data belonging to one company[3].

[1] She needs the right access only.

[2] In contrast to [2] and [3] we include the behaviour of the security manager in the model and restrict his possibilities.

[3] In fact the model does not distinguish between different data of the same company. This is necessary only if there are data with different security labels as in [1] and [4]. We do not

Of course, data can be grouped together in different ways depending on the application. Consider, for example, a big company working in different areas, e.g. producing computers and refrigerators. Then one could divide the data of the big company in two objects according to these areas. Since there are companies which produce only one of both products the different parts of the company have different competitors. Thus a consultant might work for the big company on the subject of refrigerators and for some other computer client as well. We can integrate this in our model by viewing each part of the big company as a separate object. This seems to be reasonable because the big company itself wants to ensure confidentiality and will not copy data from one object to another object. In our model we only consider the handling of data that are already *in* the system. The only way of integrating *new* data into the system is that the companies write the data in their own object. One could integrate this information flow by giving the companies the read and write access exactly to their own datasets. But since this is independent of the Chinese Wall policy we ignore this aspect.

We introduce a special object o_0 which contains data readable for everybody, e.g. public known data or sanitized statistical data from the different companies. Only the security manager s_0 can get write access to o_0. We assume that he copies data from an object o to o_0 only in agreement with the owner of the object o. Thus the data of the objects $o \neq o_0$ are regarded as sensitive and all public data of the companies are contained in o_0.

2.2 Conflict of interest relation

Let CIR be a non-empty, symmetric relation on $O \times O$, called the *conflict of interest relation*. The idea behind this concept is that two objects are related by CIR if they represent competing companies. Therefore the relation is assumed to be symmetric but not reflexive and thus not transitive. The *reflexive closure* RC of CIR is defined by

$$(o, o') \in RC \iff (o, o') \in CIR \lor o = o'$$

In general, even the reflexive closure will not be transitive. Imagine, for example, two local companies working in different areas and thus not in conflict of interest with each other and a third global company working in both areas.[4]

The conflict of interest relation induces for $o \in O$ a *conflict area*:

$$c(o) := \{o' \mid (o, o') \in CIR\}$$

and a *neighbourhood*

$$rc(o) := \{o' \mid (o, o') \in RC\} \;=\; c(o) \cup \{o\}$$

consider such a multilevel context but it could be easily integrated.

[4]This aspect was pointed out by Lin [3]. Brewer and Nash [2] implicitly assumed a transitive reflexive closure.

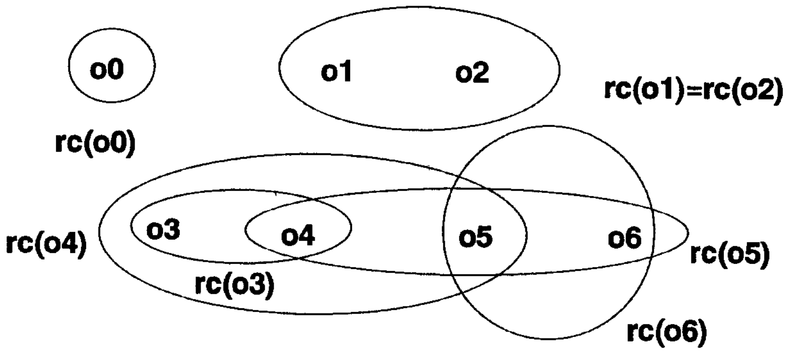

Figure 2: A system of neighbourhoods

Since the public file o_0 is assumed to be not in conflict of interest relation with any other object we have $c(o_0) = \emptyset$. Figure 2 gives an example for a system of neighbourhoods.

2.3 Access Rights

We consider two access rights read and write (abbreviated by `r` and `w`). Write access does not necessarily include read access.

Furthermore, we introduce an *access control matrix* A which maps elements of $S \times O$ to the set $\{-1, 0, 1\}$.

The interpretation of the entries is as follows:

-1	no access possible
0	not yet decided (access can be given)
1	subject has or had read access to the object

The number -1 stands for a negative decision, the number 1 stands for a positive decision and the number 0 is neither positive nor negative, which means that the subject has still free choice[5].

2.4 Requests

There are four kinds of requests

- `get_read`(s, o): Subject s wants to get read access to object o
- `release_read`(s, o): s wants to release his read access to o
- `get_write`(s, o): s wants to get write access to o
- `release_write`(s, o): s wants to release his write access to o

[5]Notice that we reversed the meaning of 0 and -1 from [3]

2.5 System States

We define the *set of access rights* as a subset $b \subset S \times O \times \{\mathtt{r}, \mathtt{w}\}$ with

$$(s, o, \mathtt{r}) \in b \iff s \text{ has read access to } o$$

$$(s, o, \mathtt{w}) \in b \iff s \text{ has write access to } o$$

Let $T = \{0, 1, 2, ..., t, ...\}$ be the set of the time indices. We start with a given conflict of interest relation CIR but we change it during the lifetime of the system in a way depending on the development of b [6]. We introduce for all $t \in T$ a relation $C(t)$ on $O \times O$ starting with $C(O) = CIR$ and define as before

$$c_t(o) = \{o' \in O \mid (o, o') \in C(t)\}$$

Then a *system state* is described by an ordered triple $(A(t), b(t), C(t))$. A *state transition* from state $(A(t), b(t), C(t))$ to the possibly modified following state $(A(t+1), b(t+1), C(t+1))$ is triggered by a request on which a decision is made according to a set of rules.

2.6 Conflict Security

An information flow from one object o to a subject s or to another object o' is possible if there is a subject s_1 who reads data from o and writes afterwards to another object o_1 and there is a subject s_2 reading o_1 and writing to o_2, and so on, until finally one ends up with s respectively o'. We define formally

Definition 1 *For all $s \in S; o, o' \in O; t, t' \in T$ with $t \leq t'$ let*

1. $o \rightsquigarrow_{(t,t')} s :\iff$

$$\exists s_1, s_2, \ldots, s_k = s \in S;\ o_1, o_2, \ldots, o_{k-1} \in O;$$

$$\exists t_1, t_2, \ldots t_{2k-1} \in T \text{ with } t \leq t_1 \leq t_2 \leq \ldots t_{2k-1} \leq t' :$$

$$(s_1, o, \mathtt{r}) \in b(t_1) \land$$

$$\forall i = 1, \ldots, k-1 : (s_i, o_i, \mathtt{w}) \in b(t_{2i}) \land (s_{i+1}, o_i, \mathtt{r}) \in b(t_{2i+1})$$

2. $o \rightsquigarrow_{(t,t')} o' :\iff o = o' \quad \lor$

$$\exists s_1, s_2, \ldots, s_k \in S;\ o = o_1, o_2, \ldots, o_{k+1} = o' \in O;$$

$$\exists t_1, t_2, \ldots t_{2k} \in T \text{ with } t \leq t_1 \leq t_2 \leq \ldots \leq t_{2k} \leq t' :$$

$$\forall i = 1, \ldots, k : (s_i, o_i, \mathtt{r}) \in b(t_{2i-1}) \land (s_i, o_{i+1}, \mathtt{w}) \in b(t_{2i})$$

[6] This differs very much from [2] and [3] where the conflict of interest relation remains constant during the lifetime of the system.

The meaning of $o \rightsquigarrow_{(t,t')} s$ (resp. o') is that there has been an information flow from object o to subject s (resp. object o') within the time period (t, t') [7]. We note some elementary properties:

- For all $t'' \geq t'$: $o \rightsquigarrow_{(t,t')} s$ (resp. o') implies $o \rightsquigarrow_{(t,t'')} s$ (resp. o')
- $o \rightsquigarrow_{(t,t')} o' \iff o = o' \quad \vee$
  $$\exists s \in S; t^* \in T : t \leq t^* \leq t' \wedge o \rightsquigarrow_{(t,t^*)} s \wedge (s, o', \mathtt{w}) \in b(t^*))$$
- For fixed $t, t' \in T$ with $t \leq t'$ the relation $\rightsquigarrow_{(t,t')}$ on $O \times O$ is reflexive, but in general not symmetric and not transitive.
- It is *weakly transitive*, i.e.
  $$o_1 \rightsquigarrow_{(t_1,t_2)} o_2 \wedge o_2 \rightsquigarrow_{(t_3,t_4)} o_3 \wedge t_2 \leq t_3 \quad \Rightarrow \quad o_1 \rightsquigarrow_{(t_1,t_4)} o_3$$

We abbreviate $o \rightsquigarrow_{(0,t')} s$ (resp. o') by $o \rightsquigarrow_t s$ (resp. o').

Definition 2 *A system is called* conflict secure *iff*

$$\forall s \in S; o, o' \in O, t \in T : \ o \rightsquigarrow_t s \wedge o' \rightsquigarrow_t s \ \Rightarrow \ s = s_0 \vee (o, o') \notin CIR$$

3 Rules

3.1 Read Access

Rule 1 *Initialization*

$$\begin{aligned} A_0(s,o) &= \begin{cases} 1 & : \ s = s_0 \vee o = o_0 \\ 0 & : \ s \neq s_0 \wedge o \neq o_0 \end{cases} \\ b(0) &= \emptyset \\ C(0) &= CIR \end{aligned}$$

Rule 2 *Concerning* get_read(s, o)

1. *If* $A_t(s,o) = 1 \wedge (s = s_0 \vee o = o_0 \vee \forall o' \neq o : (s, o', \mathtt{w}) \notin b(t))$
   then $b(t+1) = b(t) \cup \{(s, o, \mathtt{r})\}$ [8]
2. *If* $A_t(s,o) = 0 \wedge \forall o' \neq o : (s, o', \mathtt{w}) \notin b(t)$ *then*
   $$\begin{aligned} b(t+1) &= b(t) \cup \{(s, o, \mathtt{r})\} \\ A_{t+1}(s,o) &= 1 \\ A_{t+1}(s,o') &= -1 \quad \forall o' \in c_t(o) \end{aligned}$$

[7] Of course, as in any other access control model there is another information flow possible by using covert channels: If a request for an access to an object is denied this denial also gives information about this object to the requestor.

[8] We only mention those parameters that change.

3. In all other cases the request is denied.

When s gets read access to o the very first time this is recorded and a Chinese Wall is built, so that s cannot get any read access to objects inside the conflict area of o. We will later explain why s is not allowed to have any write access while getting read access.

Rule 3 *Concerning* `release_read`(s, o)
Set $b(t+1) = b(t) \setminus \{(s, o, \mathtt{r})\}$ [9]

Although the subject s cannot read o any more the entry $A(s, o)$ is not changed to 0 because s is assumed to have a memory and may still keep the data read from o. For sake of simplicity we make even a stronger assumption:

Conservative Assumption If $A_t(s, o) = 1$ then subject s will request read access to object o again and again.

This seems to be plausible because the Chinese Wall is already built and therefore a second read access to the same object will not cause any further restriction on the choice of the subject[10].

3.2 Write Access

Rule 2 prevents direct information flows from competing objects to the same subject. But it does not prevent indirect information flows arising from the write access as shown in figure 1.

Brewer and Nash [2] introduced a rule which grants write access $(s, o, \mathtt{w})$ if and only if (in our notation)

$$A_t(s, o) = 1 \wedge \forall o' \in O \setminus \{o, o_0\} : A_t(s, o') \neq 1$$

The first condition allows write access only to objects the subject has read before. The second condition implies that a subject can get write access in the beginning only: As long as he has had read access to one object $o \neq o_0$ he can write to this object. But as soon as he has had read access to a second object $o' \neq o, o_0$ he will never get any write access at all [11].

Lin [3] stated this rule as follows:

$$A_t(s, o) \neq -1 \wedge \forall o' \in O, o' \neq o, o_0 : A_t(s, o') = -1$$

Obviously, Lin was not aware of changing the second condition by translating it in his notation. He made the restriction even worse: $A_t(s, o') = -1$ means that there is an object $o'' \in c(o)$ with $A_t(s, o'') = 1$. Therefore Lin's second condition can only be satisfied if all objects $o' \neq o, o_0$ lie in the conflict area of o in which

[9] This is still correct if s did not have read access to o in state t.
[10] Note that this assumption is more or less implicitly included in [2] and [3].
[11] Surprisingly, Brewer and Nash do not mention this strong consequence.

case the whole scenario is very trivial. Thus this rule implies that in general there would be no write access at all!

The problem with the write access is that as soon as a subject s has had read access to an object o and gets write access to a second object o' there is an information flow $o \rightsquigarrow_t o'$ and therefore the other subjects cannot be allowed to read both an object $o'' \in c(o)$ and o'. Although, first of all, the two objects o' and o'' were not in conflict of interest there is now a conflict induced by the information flow $o \rightsquigarrow_t o'$. As a consequence the other subjects are restricted in their choice because of the action of s.

Giving write access in a reasonable and sufficient way implies that the actions of subjects have influence on the free choice of the other subjects which means a sort of denial of service for the individual subject.

At this point one has to decide whether to accept such influence or not. If no influence is accepted one has to take the write-rule of Brewer and Nash with the consequence that there is almost no write access at all. Thus the information flow in the model is only unidirectional, from the objects to the subjects.

In this paper we are trying the other way. If subject s gets write access to object o in state $t+1$ he could copy data from all the objects o' he read before, i.e. those objects with $A_t(s,o') = 1$. Therefore the object o gets into conflict of interest relation with all the conflict areas $c(o')$. We define

$$c_{new} := c_{new}(s,o,t) := \bigcup_{\{o' \neq o \mid A_t(s,o')=1\}} c_t(o')$$

Rule 4 *Concerning* $\mathtt{get\_write}(s,o)$
Case $s \neq s_0$ and $o \neq o_0$: The request is granted iff

$$\begin{aligned} & A_t(s,o) \neq -1 && (1)\\ \wedge\ & \forall s' \neq s, s_0 : \ (s',o,\mathtt{r}) \notin b(t) && (2)\\ \wedge\ & \forall s' \neq s, s_0 \ \mathit{with}\ A_t(s',o) = 1 : \forall o' \in c_{new} : \ A_t(s',o') \neq 1 && (3) \end{aligned}$$

In the positive case the state is modified as follows

$$b(t+1) = b(t) \cup \{(s,o,\mathtt{w})\}$$

Changing of C by:

$$\begin{aligned} & c_{t+1}(o) = c_t(o) \cup c_{new} && (4)\\ & \forall o' \in c_{new} : \ c_{t+1}(o') = c_t(o') \cup \{o\} && (5) \end{aligned}$$

Building Chinese Walls:

$$\begin{aligned} & \forall s' \neq s, s_0 \ \mathit{with}\ A_t(s',o) = 1 : \forall o' \in c_{new} : \ A_{t+1}(s',o') = -1 && (6)\\ & \forall o' \in c_{new} \forall s' \neq s, s_0 \ \mathit{with}\ A_t(s',o') = 1 : \ A_{t+1}(s',o) = -1 && (7) \end{aligned}$$

Case $s = s_0$ or $o = o_0$: The request is granted iff $s = s_0$ and $o = o_0$. Then

$$b(t+1) = b(t) \cup \{(s,o,\mathtt{w})\}$$

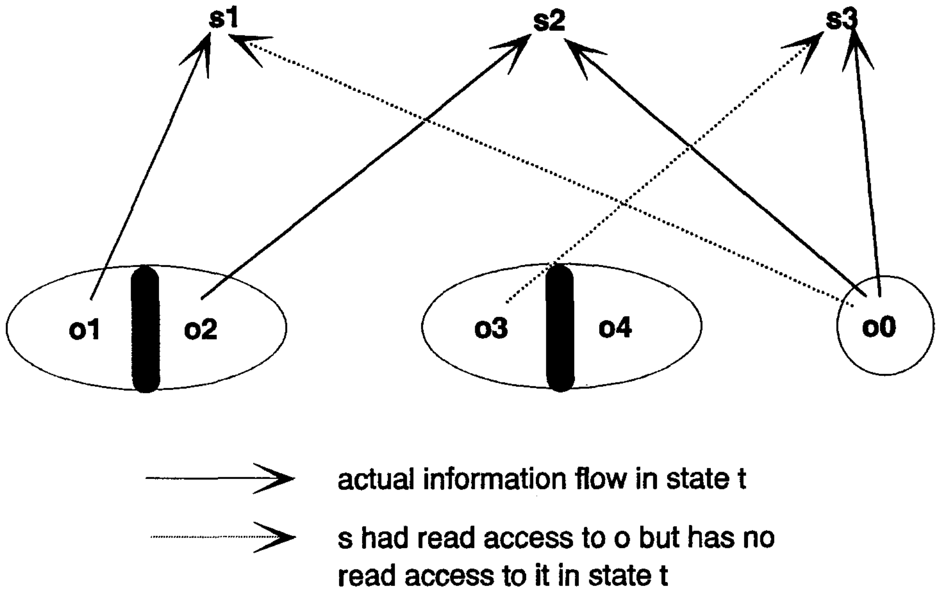
s1
s2
s3
o1
o2
o3
o4
o0
actual information flow in state t
s had read access to o but has no read access to it in state t

Figure 3: Applying rule 4 to $\texttt{get\_write}(s_1, o_3)$

Example Suppose there are three subjects s_1, s_2, s_3 and five objects $o_0, o_1, \ldots, o_5$, see figure 3. For sake of simplicity we ignore s_0. The state t is described by the following parameters:

$$\begin{aligned}
A(t) &= \begin{pmatrix} 1 & 1 & -1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & -1 \end{pmatrix} \\
b(t) &= \{(s_1, o_1, \mathtt{r}), (s_2, o_0, \mathtt{r}), (s_2, o_2, \mathtt{r}), \\
&\quad (s_3, o_0, \mathtt{r}), (s_3, o_3, \mathtt{r})\} \\
C(t) &= \text{symmetric closure of } \{(o_1, o_2); (o_3, o_4)\}
\end{aligned}$$

Suppose now that s_1 requests write access to o_3. Then

$$c_{new} = \bigcup_{\{o' \neq o | A_t(s_1, o') = 1\}} c_t(o') = c_t(o_0) \cup c_t(o_1) = \emptyset \cup \{o_2\}$$

Since $A_t(s_1, o_3) = 0$ and no subject reads o_3 the conditions (1) and (2) are fulfilled. Checking condition (3) we notice that s_3 is the only subject who had read access to o_3, $c_{new} = \{o_2\}$, and $A_t(s_3, o_2) = 0 \neq 1$. (Note that if we had $A_t(s_2, o_3) = 1$ this condition would not be satisfied.) The write request is granted and the modified parameters are

$$A(t+1) = \begin{pmatrix} 1 & 1 & -1 & 0 & 0 \\ 1 & -1 & 1 & -1 & 0 \\ 1 & 0 & -1 & 1 & -1 \end{pmatrix}$$

$$\begin{aligned} b(t+1) &= \{(s_1,o_1,\mathtt{r}),(s_1,o_3,\mathtt{w}),(s_2,o_0,\mathtt{r}),\\ &\quad (s_2,o_2,\mathtt{r}),(s_3,o_0,\mathtt{r}),(s_3,o_3,\mathtt{r})\}\\ C(t+1) &= \text{symmetric closure of } \{(o_1,o_2);(o_2,o_3);(o_3,o_4)\} \end{aligned}$$

Remarks on the condition (2): At first glance, it seems to be a very restrictive condition that a subject only can write to an object if no other subject has read access to this object. But, in fact, most common data base systems today have this condition anyway in order to ensure the consistency of the data. To ensure conflict security it would suffice to demand

$$\forall s' \neq s, s_0 : \ (s', o, \mathtt{r}) \in b(t) \ \Rightarrow\ \forall o' \neq o : \ (s', o', \mathtt{w}) \notin b(t)$$

But the stronger condition (2) has the further advantage that it only concerns the object o and not the other activities of the subjects s' reading o.

Rule 2 forbids a subject to have write access to any other object o' while gaining read access to o. There would be an alternative to check more exactly to which objects the subject has write access and where the information might go on. But our philosophy was to keep rule 2 as simple as possible and to put the cumbersome checks into rule 4 because the write access is responsible for the indirect information flow. Therefore we decided against this alternative.

Rule 4 is implicitly influenced by our conservative assumption which we formulate more precisely now.

Conservative Axiom For all $s \in S, o \in O, t \in T$:

$$(s,o,\mathtt{w}) \in b(t) \Rightarrow \forall o' \in O \text{ with } A_t(s,o') = 1 : \ (s,o,\mathtt{r}) \in b(t)$$

This means that a subject only has write access if he has read access to all objects he read before. The idea behind this axiom that a subject should always have the opportunity to get read access to objects he read before. The axiom is not really necessary in this strength but it simplifies some formulas. Especially, we have

$$\begin{gathered} o \rightsquigarrow_{(t,t')} o' \iff o = o' \ \vee \\ \exists s_1, s_2, \ldots, s_k \in S;\ o = o_1, o_2, \ldots, o_{k+1} = o' \in O; \\ \exists t_1, t_2, \ldots t_k \in T \text{ with } t \leq t_1 \leq t_2 \leq \ldots t_k \leq t' : \\ \forall i = 1, \ldots, k : \ A_{t_i}(s_i, o_i) = 1 \ \wedge\ (s_i, o_{i+1}, \mathtt{w}) \in b(t_i) \end{gathered}$$

In terms of the original definition 1 we replaced the condition $(s_i, o_i, \mathtt{r}) \in b(t_{2i-1})$ by $A_{t_{2i}}(s_i, o_i) = 1$.

Rule 5 *Concerning* `release_write`(s,o)
Set $b(t+1) = b(t) \setminus \{(s,o,\mathtt{w})\}$

4 Theorems

Theorem 1 *For all $o, o' \in O; t \in T$:*

$$\begin{array}{lll} & (o,o') \in C(t+1) & \\ \Longleftrightarrow & (o,o') \in C(t) & \\ \vee & \exists s \in S: \ (s,o,\mathbf{w}) \in b(t+1) \setminus b(t) \ \wedge \ \exists o'' \in c_t(o') \ A_t(s,o'') = 1 & (8) \\ \vee & \exists s \in S: \ (s,o',\mathbf{w}) \in b(t+1) \setminus b(t) \ \wedge \ \exists o'' \in c_t(o) \ A_t(s,o'') = 1 & (9) \\ \Longleftrightarrow & (o,o') \in C(t) & \\ \vee & \exists o'' \in c_t(o'): \ o'' \leadsto_{(t+1,t+1)} o & \\ \vee & \exists o'' \in c_t(o): \ o'' \leadsto_{(t+1,t+1)} o' & \end{array}$$

Proof: C is only changed when rule 4 is applied. Formula (4) of rule 4 corresponds to formula (8) and formula (5) corresponds to formula (9). The other equivalence follows from the definition of $\leadsto_{(t+1,t+1)}$ and the Conservative Axiom.

Theorem 2 *For all $t \in T$:*
1. $C(t)$ is a symmetric relation on $O \times O$
2. $c_t(o_0) = \emptyset$
3. $C(0) \subset C(1) \subset \ldots \subset C(t) \subset \ldots$

Proof: This follows by induction from theorem 1 because the initial relation $C(0) = CIR$ is symmetric and $c(o_0) = \emptyset$.

Note that even if the reflexive closure RC(0) is transitive in general the reflexive closure RC(t) is not transitive because the modifications do not preserve transitivity.

Theorem 3 *For all $o, o' \in O; t \in T$:*

$$(o,o') \in C(t) \ \Rightarrow \ \exists (p,p') \in CIR: p \leadsto_t o \ \wedge \ p' \leadsto_t o'$$

Proof: By induction on t. The case $t = 0$ is trivial: Set $(p,p') = (o,o')$.

Assume it holds for t and let $(o,o') \in C(t+1)$. Applying theorem 1 we consider the first case $(o,o') \in C(t)$: Then the induction hypothesis yields the existence of $(p,p') \in CIR$ with $p \leadsto_t o$ and $p' \leadsto_t o'$ and thus $p \leadsto_{t+1} o$ and $p' \leadsto_{t+1} o'$.

Consider now the second case of theorem 1 (which is analogous to the third case): Then there exists $o'' \in c_t(o')$ with $o'' \leadsto_{(t+1,t+1)} o$. According to the induction hypothesis there exists $(p,p') \in CIR$ with $p \leadsto_t o''$ and $p' \leadsto_t o'$ and therefore $p \leadsto_{t+1} o$ and $p' \leadsto_{t+1} o'$.

Theorem 4 *For all $s \in S \setminus \{s_0\}, o \in O, t \in T$:*

$$A_t(s,o) = -1 \iff \exists o' \in c_t(o) : A_t(s,o') = 1$$

Proof: From left to right: Without restriction let t be the state where $A_t(s,o)$ was changed to -1. First case: The state transition from $t-1$ to t was triggered by **get_read**(s',o'). But then it must be $s = s'$ and $o' \in c_{t-1}(o)$.

Second case: The new state results from applying rule 4 to **get_write**(s',o') which implies $s' \neq s$.

Subcase $o' = o$: Then the formula (7) of rule 4 was applied and there exists $o'' \in c_{new} \subset c_t(o)$ with $A_{t-1}(s',o'') = 1$.

Subcase $o' \neq o$: Then the formula (6) was applied so that $A_{t-1}(s,o') = 1$ and $o \in c_{new} \subset c_t(o')$.

From right to left: By induction on t we prove the following statement

$$A_t(s,o') = 1 \wedge (o,o') \in C(t) \Rightarrow A_t(s,o) = -1$$

The case $t = 0$ is trivial because $A_0(s,o') = 1$ implies $o' = o_0$ and $c_0(o_0)$ is empty according to rule 1. Assume it holds for t and let $A_{t+1}(s,o') = 1$ and $(o,o') \in C(t+1)$. Suppose further that either $A_t(s,o') \neq 1$ or $(o,o') \notin C(t)$ (otherwise it is trivial by induction hypothesis). Only rule 2 changes entries of A to 1 but this rule does not change C. Therefore, *exactly* one of both cases holds.

Consider the first case $A_t(s,o') \neq 1$ and $(o,o') \in C(t)$. Then the state transition to $t+1$ was triggered by **get_read**(s,o'). Since $o \in c_t(o')$ rule 2 yields $A_{t+1}(s,o') = -1$.

Consider now the other case $A_t(s,o') = 1$ and $(o,o') \notin C(t)$. Then theorem 1 implies that a write request must have triggered the state transition to state $t+1$ and that either formula (8) or formula (9) holds.

If (8) holds there exist a subject s' getting write access to o and an object $o'' \in c_t(o')$ with $A_t(s',o'') = 1$ which implies $o' \in c_{new}$. Since $A_t(s,o') = 1$ we conclude by induction hypothesis that $A_t(s,o'') = -1$ and thus $s \neq s'$. Then by formula (7) of rule 4 the entry $A(s,o)$ is changed to -1.

If formula (9) holds there exist a subject s' getting write access to o' and an object $o'' \in c_t(o)$ with $A_t(s',o'') = 1$ implying $o \in c_{new}$. If $s' \neq s$ the entry $A_{t+1}(s,o')$ is changed to -1 by applying formula (6). If $s' = s$ the induction hypothesis applied to $(o,o'') \in C(t)$ yields already $A_t(s,o) = -1$.

The next two theorems correspond to two basic properties of the Bell-LaPadula model [1]: The Simple Security Property "no read up" which restricts read access and the *-Property which restricts the flow of information between different objects.

Theorem 5 *(Simple Security Property)*
For all $s \in S \setminus \{s_0\}, o, o' \in O, t \in T$:

$$A_t(s,o) = 1 \wedge A_t(s,o') = 1 \Rightarrow (o,o') \notin C(t)$$

Proof: This follows directly from theorem 4 "$\Leftarrow$": If $A_t(s,o) = 1$ and $(o,o') \in C(t)$ the entry of $A_t(s,o')$ is -1 instead of 1.

Theorem 6 *(*-Property)*
For all $o, o' \in O; t, t' \in T$ with $t \leq t'$:

$$o \rightsquigarrow_{(t,t')} o' \;\Rightarrow\; c_t(o) \subset c_{t'}(o')$$

Proof: This follows directly from theorem 1 by induction on the number k of subjects in the path from o to o'.

Theorem 7 *(Inversion of theorem 3)*
For all $o, o', p, p' \in O; t \in T$:

$$(o,o') \in CIR \wedge o \rightsquigarrow_t p \wedge o' \rightsquigarrow_t p' \;\Rightarrow\; (p,p') \in C(t)$$

Proof: Without restriction we can assume that there exists $t' \leq t$ with $o \rightsquigarrow_{t'} p$ and $o' \rightsquigarrow_{(t',t)} p$. Then we conclude by theorem 6 that $o' \in c_{t'}(p)$ which is equivalent to $p \in c_{t'}(o')$. Again by theorem 6 we conclude $p \in c_t(p')$.

Theorem 8 *The system is conflict secure, i.e. for all $s \in S \setminus \{s_0\}; o, o' \in O; t \in T$:*

$$o \rightsquigarrow_t s \wedge o' \rightsquigarrow_t s \Rightarrow (o,o') \notin CIR$$

Proof: Let $o \rightsquigarrow_t s$ and $o' \rightsquigarrow_t s$. There exist $p, p' \in O$ with $o \rightsquigarrow_t p$, $o' \rightsquigarrow_t p'$, $A_t(s,p) = 1$, and $A_t(s,p') = 1$. We conclude by theorem 5 that $(p,p') \notin C(t)$ and thus by theorem 7 that $(o,o') \notin C(0) = CIR$.

5 Final Remarks

A problem about our approach is that there are a lot of Chinese Walls built but none is erased again. This is a consequence of our conservative axiom. Of course, one might argue about this axiom because one can imagine situations in which subjects will definitely release read access to certain objects. Then, after a period of time, the subject does not have any more insider knowledge from this object because everything he read is now publicly known anyway. Therefore the matrix A could be updated so that the subject can get read access to objects inside of the conflict area of the first object, i.e. the Chinese Wall is erased again. In our approach this Chinese Wall might have be responsible for other Chinese Walls built throughout the system by applying rule 4. Therefore the system manager must audit every action and save these audit files for a certain period so that he can decide which Chinese Wall can be erased. We do not want to get into details because the rules and theorems would become even more complicated although they are still manageable. In our opinion further work on this subject is only useful if one has realistic applications.

Acknowledgement

Work on this paper was done under the national project REMO funded by the German Federal Ministry of Research and Technology (BMFT) under no. 01 IS 202 E. Cooperative partners of REMO are E.I.S.S. (University of Karlsruhe), GMD, IABG mbH, Siemens AG, and TELES GmbH. Special thanks to my colleague Sibylle Mund for helpful discussions on this topic. Thanks to Esther Amann (IABG) and Birgit Klein (E.I.S.S.) for comments on this paper.

References

[1] D.E. Bell, L.J. LaPadula, *Secure Computer Systems: Unified Exposition and Multics Interpretation,*
NTIS AD-A023 588, MTR 2997, ESD-Tr-75-306, MITRE Corporation, Bedford, MA, 3/1976

[2] D.F.C. Brewer, M.J. Nash, *The Chinese Wall Security Policy,*
Proc. of the IEEE Symp. on Security and Privacy 1989, 206-214

[3] T. Lin, *Chinese Wall Security Policy - an Aggressive Model,*
Proc. of the 5th Aerospace Computer Security Conference 1989, IEEE Comp. Sec. Press 1990, 282-289

[4] C. Meadows, *Extending the Brewer-Nash Model to a Multilevel Context,*
Proc. of the IEEE Symp. on Security and Privacy 1990, 95-102