

Security Models

Andrea Burattin

From an article of John McLean (1994)

Introduction

The term *security model* can be used in two correlated, but distinct, senses:

- a. a mechanism for enforcing confidentiality, aka “access control”;
- b. specifications of a system’s requirements for confidentiality, interfaces for enforcing confidentiality.

Access control models

The model is represented as a **state machine**¹, where each state is a triple (S, O, M) where

S is a set of subject,

O is a set of object,

M is a matrix, with one row for each subject and a column for each object; $M[s, o]$ contains the rights of s for o . These rights are taken from a finite set called A .

States are changed by altering the access matrix M . An individual machine (i.e. the set of all the states) is called system.

The Harrison, Ruzzo, Ullman Model

The HRU Model, first of all, makes the problem precise, containing requests only in the following form:

```
if a1 in M[s1, o1] and
    a2 in M[s2, o2]
    ...
then op1
    op2
    ...
```

Where, each a is in A and each op is one of

```
enter a into(s, o)
delete a from(s, o)
create subject s
create object o
destroy subject s
destroy object o
```

¹A state machine is a model of behavior composed of states, transitions and actions. - From *Wikipedia*

Given a system, an initial configuration Q_0 , a right a , we say Q_0 is **safe** for a if there is no sequence of requests that will write a in a cell of M that didn't already contain it.

HRU Theorem 1 *The safety problem is decidable for mono-operational systems (where each request has only one operation), so there is an algorithm for determining if a system is safe for a right a .*

Proof: From every sequence of requests that leaks a , there is another sequence of requests that also leaks a , but contains only **enter** requests except for an initial **create subject**. We obtain a W_1 sequence = **create subject** $sx + W^*$, where W^* is created from the original sequence by dropping:

- all the **destroy/delete** (this operation is allowed: the requests test the presence of the right, not absence),
- all the **create subject** sy (replacing every test on sy with sx).

We can decide the security of a system by looking at all possible sequences **enter** of length $\leq (|A| \cdot (|S_0| + 1) \cdot (|O_0| + 1)) + 1$.

HRU Theorem 2 *The general problem of determining if a given situation of a given system is safe for a right a is undecidable.*

Proof: We can reduce the problem of the safety of a system to the problem of the halting of a Turing Machine², so we are done. To simulate this reduction consider $A + statek$ as the matrix right, the final state qf [of the TM] as the right p of the problem and the Turing Machine tape as the diagonal of the access matrix.

The Bell and LaPadula Model

This model differs from HRU one in that the set S and O don't change from state to state and the set A contains only two rights (*read*, *write*). Another difference is that BLP model introduces a security level, called L and a function $F : S \cup O \rightarrow L$. The set of states V is a set of pairs (F, M) where M is the access matrix. A system consists of an initial state v_0 , a set R of requests and a transition function $T : (U \cdot R) \rightarrow V$ that transform the system. There are also a series of definitions that are necessary and sufficient criteria for a system, to be secure (this is only a *definition*):

- a. **SS-Property:** a state is “**read secure**” if from every $s \in S$, $o \in O$, $read \in M[s, o] \rightarrow F(s) \geq F(o)$, aka no read up;
- b. ***-Property:** a state is “**write secure**” if from every $s \in S$, $o \in O$, $write \in M[s, o] \rightarrow F(o) \geq F(s)$, aka no write down;
- c. a state is *secure* if it is “read” and “write secure”;
- d. a system is secure if v_0 (the starting state) is secure and every state reachable from v_0 is secure.

In this scenario we observe that the information can only grow their security level and this means that the confidentiality is guaranteed.

²Turing machines are extremely basic symbol-manipulating devices which can be adapted to simulate the logic of any computer that could possibly be constructed. - From *Wikipedia*

BLP Theorem 1 A system (v_0, R, T) is secure if:

1. v_0 is a secure state;
2. T is such that for every state reachable from v_0 , executing a finite sequence of requests $\in R$ if and only if $T(v, c) = v^*$ where $w = (F, M)$ and $v^* = (F^*, M^*)$, $\forall s \in S$ and $\forall o \in O$:
 - a) if $read \in M^*[s, o]$ and $read \notin M[s, o]$ then $F^*(s) \geq F^*(o)$;
 - b) if $read \in M[s, o]$ and $F^*(s) \not\geq F^*(o)$ then $read \notin M^*[s, o]$;
 - c) if $write \in M^*[s, o]$ and $write \notin M[s, o]$ then $F^*(o) \geq F^*(s)$;
 - d) if $write \in M[s, o]$ and $F^*(o) \not\geq F^*(s)$ then $write \notin M^*[s, o]$;

Proof (“only if”): if v_0 is not secure, we are done. If we can reach v^* , it must satisfy both the (a) and (b) other case v^* is not *read secure*, it must satisfy also (c) and (d) to be *write secure*.

(“If”): if the system is not secure we have that v_0 is not secure, or a state reachable v^* must be a non-secure state (for *read* or *write*), so the ((a and b) or (c and d)) failed.

Problem with BLP model

Consider the system Z whose initial state is secure and which has only one type of transition: when every $s \in S$ request one of every $o \in O$, s and o are downgraded to the lowest level of L so access is granted, **always**. Z satisfies BLP’s notion of security but, it’s absolutely unsafe. We must define restrictions to have only transition secure.

BLP’s McLean derivative

A framework is a quadruple (S, O, L, A) as in BLP model but there is also a new function $C : S \cup O \rightarrow P(S)$ that returns the set of s allowed to change security level. The new transition function is $T : (S \cdot V \cdot R) \rightarrow V$. Now we can define:

- a transition function is secure if and only if $T(s, v, r) = v^*$ where $v = (f, m)$ and $v^* = (f^*, m^*)$, implies that $\forall x \in S \cup O$, if $f(x) \neq f^*(x)$ then $s \in C(x)$;
- a system is secure if v_0 and all reachable states are secure and T is a secure transition.

Any case we have to define C , but this is not a problem: we can decide it “moving” from $C(x) = S$ for a “traditional BLP” (with T that accepts every R) or $C(x) = \emptyset$ and having intermediate politics. If $C(x) = \emptyset$ no one can modify F : this situation is called *tranquility*.

References

- [1] John McLean: *Security Models*, (1994)
- [2] Gilberto Filè: Slides del corso “Sicurezza nei sistemi di calcolo”, Università degli Studi di Padova (2006)