Row-level MAC in H2

Christopher Martin
chris.martin@gatech.edu

December 7, 2013

Abstract

This project explores how a relational database manager can be modified to enforce a mandatory access control policy (MAC) using a multi-level security (MLS) model. As a prototype, I modified the H2 DBMS to enforce read permissions on a per-tuple granularity and include security-related additions to the SQL grammar.

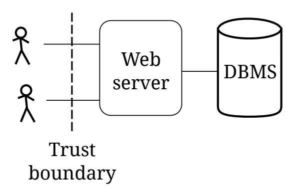

Figure 1: A common web architecture with a web application server and DBMS in the same trust domain.

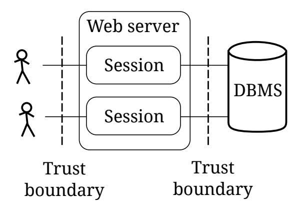

Figure 2: Architecture one in which the DBMS has sufficient power to distrust its clients, enforcing an additional trust boundary.

1 Motivation

The aim is to help provide strong guarantees for information protection in information systems. In particular I consider the requirements of U.S. Department of Defense information systems, but the result is conceivably generalizable to other domains, notably for the protection of patient privacy in healthcare systems.

As a typical example of an information system architecture, consider a mostly-stateless web application using a relational database for persistence, depicted in Figure 1. In this picture, the web application connects to a database with permissions sufficient to access data for all users of the system.

Instead we strive for the alternative of Figure 2. In this design, each user's session corresponds to a dedicated database connection associated with that user's credentials. Even if the server suffers from a SQL injection vulnerability that allows a user to execute arbitrary queries, since the database itself is responsible for enforcing access control, the user is unable to gain unauthorized access. The web server is still a trusted component of the system, but the complexity of its burden has been largely reduced to simply avoiding crosstalk between sessions.

Oracle has a similar extension to Oracle Database called Oracle Label Security (OLS)[3].

2 Security model

Here we consider a slightly simplified variant of the multilevel security (MLS) model utilized in U.S. Department of Defense information systems.[1]

Sensitivity level A fully ordered set of labels. Herein we call these labels 0, 1, 2, and 3, where the infimum level 0 denotes the least sensitive data, and the supremum level 3 is applied to the most highly-guarded secrets.

Compartment A set of labels indicating topics to which data pertains. This set is partially ordered because topics may be nested hierarchically. For example, if the compartment *Food* contains the compartments *Apples* and *Oranges*, then *Food* > *Bananas*, but *Apples* and *Oranges* are incomparable.

Marking A marking consists of one sensitivity level and one or more compartments. We denote this by separating the components with slashes, such as *2/Apples/Bananas*. *0* is a special case for which no compartments are permitted, because that sensitivity represents no access control whatsoever. Markings form a lattice where the infinum is *0* and the supremum is a marking with the highest sensitivity and every compartment.

Credential A credential consists of a sensitivity level and a compartment. Each subject has a set of credentials.

A subject has read access to data marked by $\ell/C_1/C_2/\ldots/C_n$ only if for all $i \in [1, n]$ the subject has a credential (ℓ', C') such that $\ell' \geq \ell$ and $C' \geq C_i$.

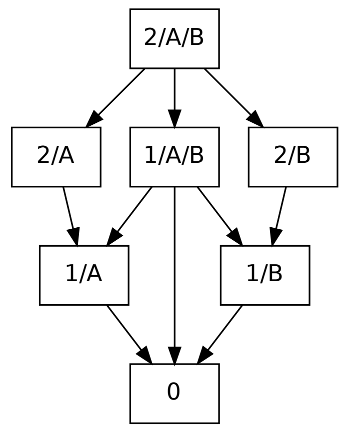

Figure 3: The lattice of markings formed using sensitivities $\{0, 1, 2\}$ and compartments $\{A, B\}$. Arrows point from more restrictive to less restrictive markings.

3 Implementation

3.1 The `MAC` schema

When a new database is initialized, it creates a schema called `MAC` whose tables are given by Figure 4. This stores sensitivities, compartments, markings, and users' credentials. The most critical use

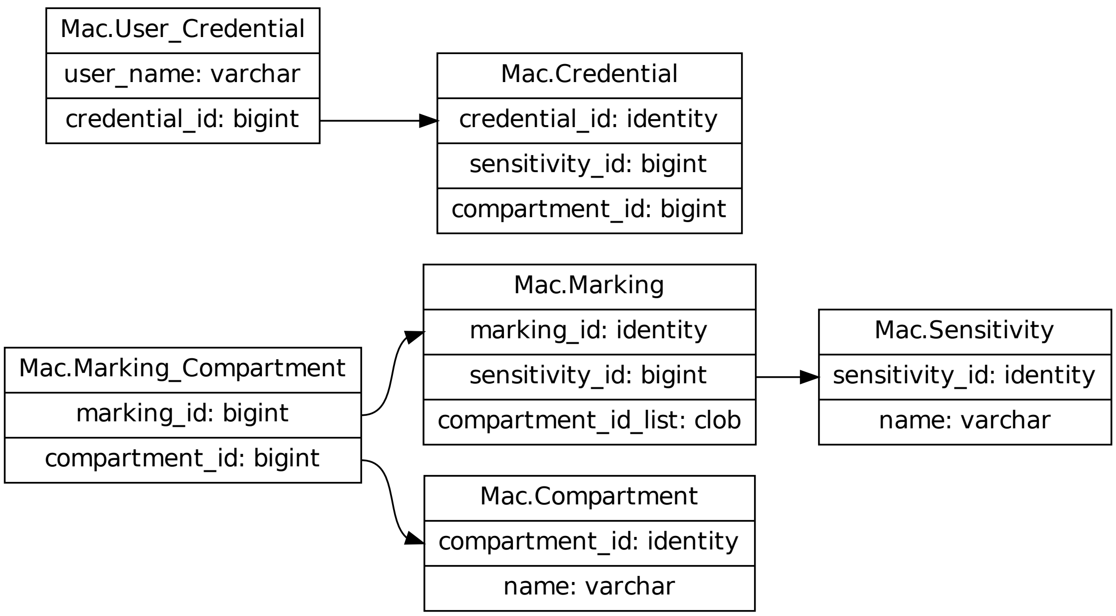

Figure 4: The tables of the underlying `MAC` schema.

of these tables is to generate a table view in the MAC schema called session_marking, a single-column relation containing an entry corresponding to each marking that the current session is permitted to access.

3.2 SQL grammar modifications

- When creating a new schema, the keyword restricted enables the security feature.

  ```
  create restricted schema "schema_name";
  ```

- The grant syntax is modified to allow granting MLS credentials to users.

  ```
  grant marking '3/B' to alice;
  ```

- The insert syntax is modified to allow specification of a marking,

  ```
  insert into people marked '2/B/C' (name)
  values ('Bob');
  ```

3.3 Query interpretation

When a user creates a restricted schema,

```
create restricted schema vault;
```

the database actually creates two schemas.

```
create schema vault;
create schema vault_shadow;
```

The "shadow" schema contains all of the relations created explicitly by create table statements, and the other schema contains restricted views into those tables.

When we create a table called doc on the vault schema,

```
create table vault.doc (title
varchar(20))
```

the database actually adds the table to the shadow schema, and places a restricted view into the visible schema.

```
create table vault_shadow.doc (
  title varchar(20),
  marking_id bigint
);

create view vault.doc as
  select title,
  render_marking(marking_id)
  from vault_shadow.doc
  join mac.session_marking;
```

Finally, when the database executes an insert query,

```
insert into vault.doc
marked '2/A' (title) values
('hi');
```

the database translates this into an insertion into the shadow table, with the marking string parsed and converted to a marking identifier.

```
insert into
vault_shadow.doc marked
(title, marking_id) values
('hi', ...);
```

4 Performance evaluation

The test setup models a system for storing multi-page documents. The Document and Page tables are both separately protected with markings, so a user who can see a document may not be able to see all of its pages.

Figure 6 shows how query execution time varies as a function of the size of the Document relation. Using an on-disk database, the access control incurs a cost of roughly 37 milliseconds per document. H2 can also run purely in memory, in which can the penalty is only 3 milliseconds.

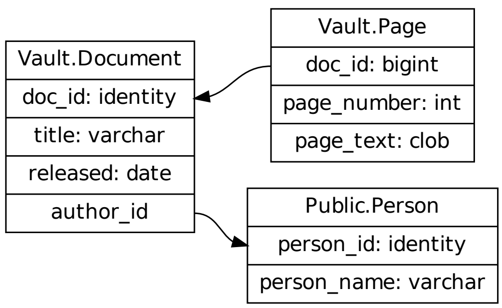

Figure 5: The tables in the schemata used for testing.

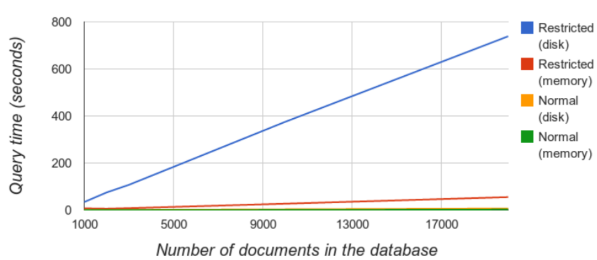

Figure 6: Execution time for a query which joins all three tables.

References

[1] Department of Defense. 1985. *Trusted Computer System Evaluation Criteria* (Orange Book). “Division B: Mandatory protection”

[2] H2 database. `http://h2database.com/`

[3] Oracle Label Security. `http://www.oracle.com/us/products/database/options/label-security/overview/index.html`