

Meta-Data switching - High Performance Data Security filtering

Mark Stocks and David Finlayson
Director Designing IT Solutions
marks@designingITS.com

Abstract- The primary enterprise risk mitigation mechanisms are the perimeter defenses, the DMZ and role-based security. Role-based security is the dominant enterprise mechanism to restrict a user's access to data by routing the user to the correct process or application. There are a number of enterprise weaknesses in the implementation of role-based security. We introduce our meta-data switching framework, specific high performance switching algorithms that exploit mathematical properties encoded into security labels, attached to data. The switching provides row level security filtering capability. We give an example of such an algorithm to show the features of meta-data switching.

I. INTRODUCTION

The sharing of information is a key human activity and the need-to-know and the need-to-share are the two security dimensions of information that are in constant tension.

The need-to-share is the dimension of allowing others to review and contribute to some collection of data and information while the need-to-know is the balancing force of allowing a subset of people access that should have access.

Our data security consulting practice brings us into contact with numerous organisations and they all report the same class of data and security problems:

- The external hacker searching for vulnerabilities in perimeter defences
- Internal people looking at information they shouldn't have access to
- The loss of information, data breaches of information. Private and sensitive information leaving the organisation to unauthorised third parties. Sometimes accidental and sometimes not

II. THE CURRENT PROBLEM

The primary enterprise risk mitigation mechanisms are the perimeter defences, the DMZ. The construction of perimeter defences using firewall technology to keep out external hackers, viruses and malware.

In addition enterprise X500 directory technology such as Light Directory Access Protocol (LDAP) and Active Directory (AD) are used to functionally profile the valid user based on a valid set of roles.

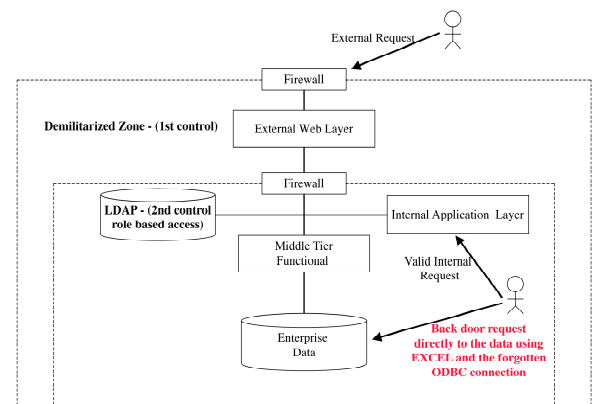
The control mechanism of role-based security [5] is the dominant enterprise mechanism to restrict a user's access to information by routing the user to the correct process or application. The application is in turn responsible for reading

and writing data (not the user). It is the application that must ensure that the user is aligned with the roles and rights assigned to them.

There are a number of weaknesses in the implementation of role-based security to protect data and information:

- There is the possibility of inappropriate back doors left open to enterprise data. The classic case is a system-based connection being created for use within an application, and then the connection being exploited by an unauthorised person. For example an unauthorised power user downloading sensitive data using a simple desktop package like Microsoft excel, connected via the ODBC connection to the database.
- The second weakness is that data security is only as good as the application written against the database. Developers are not data security specialists yet we expect them to ensure data security for user access by encoding the row level data access controls within their applications.
- Thirdly, developing security in each application requires security to be developed multiple times over the same data set where multiple applications access the same data:
 - The application approach multiplies the risk of security issues, whilst diluting the resources available to work on security.
 - It also increases the amount of both initial development, and ongoing maintenance work that needs to be performed.
 - In the situation where the actual security regime needs to change (such as when two companies merge), this can become a serious issue. In short the approach is both inefficient and risky.

BACK DOOR WEAKNESS USING ONLY ROLE BASE SECURITY ACCESS



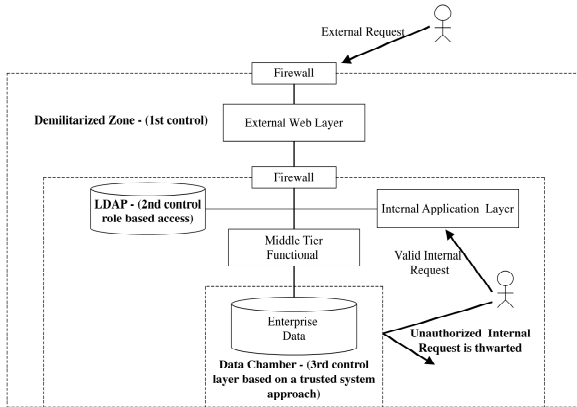
The approach of ‘application’ security is incongruent with other layers of security that are provided at the enterprise infrastructure level:

- We don’t expect the application developer to ensure that firewall like functionality is built into their application, or to open and close communication ports. The application developer can rely on communication infrastructure.
- Likewise we don’t expect the application developer to program functional routing and role based profiling as this again is provided as a layer of LDAP / AD enterprise infrastructure.

Organizations need an enterprise infrastructure approach to data security and privacy at the application level. This is no different from the approach taken in every enterprise to the other aspects of IT security. In our consultancy we have constructed such an approach that wraps around any RDBMS.

This paper introduces a meta-data switching framework and the specific mechanisms needed to overcome the role-based weaknesses as outlined above. We bring the concepts of role-based security together with the notion of a trusted system, constructing a more complete and secure model. Our approach is based on an extension of the Bell-La Padula Model [2] and the Clark-Wilson model [3] where we exploit properties of Meta-data rather than the traditional bit vector approach. We have found enormous performance gains by directly exploiting mathematical properties encoded into Meta-data.

DATA SECURITY AS A LAYER INFRASTRUCTURE AROUND THE RDBMS

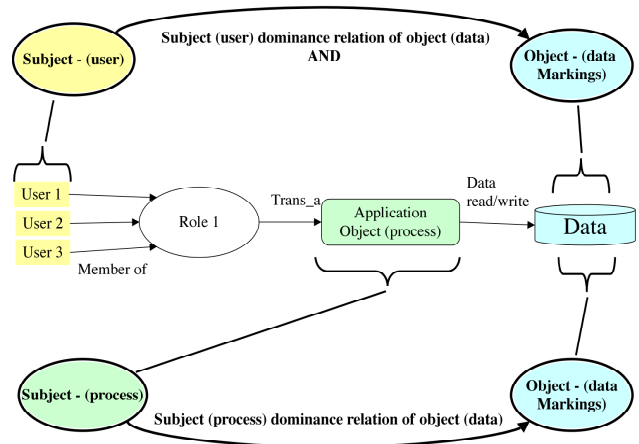


III. META DATA SWITCHING FRAMEWORK

The diagram below represents the integration of the role-based approach and our meta-data switching that we have developed. The reader will note the classical trusted system components in the integrated model:

- Subject - The security label of the active computer system entity that can initiate requests for resources. [1]
- Object – The security label of the passive computer system data resource [1] the subject wants access to.
- Dominance function - a binary relation on the set of security labels that is called dominates, such that if the Subject’s security label ‘dominates’ the Object’s security label, access is granted to the Subject.

INTERACTION BETWEEN ROLE-BASED SECURITY AND THE DOMINANCE RELATION - A COMBINED TRUSTED SYSTEM & ROLE BASED APPROACH



The accepted method of representing security meta-data is using bit-vectors. Our encoding and decoding mechanism breaks away from the traditional bit vector approach, which is low level and easy to compromise.

In contrast security labels consisting of meta-data encoding containing mathematical properties provides:

- A level of obfuscation of the actual security semantics and
- High speed switching characteristics such that row level security around the relational database is now practical.

We have patented [6] a number of algorithms and have tested some of these as a SQL wrapper around various RDBMS engines. One test comprising more than 300 million rows of data on a Teradata cluster with small sub second performance overheads being experienced to the actual time taken for the SQL queries to run.

The same algorithms could also be applied at the operating system level and could even be embedded in the RDBMS engine itself, communication devices and web servers.

A Meta-data switching framework for security has the following properties:

- There is a mathematical property about the data that is exploited and used for switching.
- Multi-Level security semantics is encoded into metadata and results in a single label.

We will demonstrate a simple Meta-data switching framework using ordinal numbers and the ‘sort-order’ functions that are found in any programming language. We will demonstrate the approach using Python code. Python is as close to executing pseudo code as can be found. It is simple to understand and easy for the reader to follow.

The example we provide using ‘sort-order’ is for demonstration purposes and is not recommended as an industrial strength solution. We have developed and patented other algorithms [6] exploiting far better mathematical properties than the simple ‘ordinal number’ property.

First we formally define security labels and the dominance function [1][2][4].

Security labels are the primary meta-data associated with either an Object of a Subject. A Security label classically is defined as follows:

$$labels = levels \times \wp(categories)$$

I.e. the set of labels is equal to a cross product of ‘the set of the levels’ and ‘the power set of categories’.

Where the level is a member of a set of classifications like ‘board’, ‘seniorExec’ or ‘staff’ such that

$$board \geq seniorExec \geq staff$$

And where a category is a member of a set of need-to-know compartments like ‘marketing’, ‘IT’, ‘production’.

For example (using a commercial example):

$$\begin{aligned} levels &= \{board, seniorExec, staff, public\} \\ category &= \{marketing, IT, production, accounts\} \\ P(categories) &= \{\emptyset, \{marketing\}, \{IT\}, \{marketing, IT\}, \dots, etc\} \end{aligned}$$

Continuing the example; three possible ordered pairs from the cross product that make up the labels include:

$$\begin{aligned} (board, \{marketing, IT\}) \\ (seniorExec, \{marketing, production, IT\}) \\ (staff, \{accounts\}) \end{aligned}$$

A binary relation on the set of labels called dominates is introduced. A subset of the cross product of $labels \times labels$:

$$subject \subseteq labels \text{ and } object \subseteq labels$$

When an ordered pair of labels ($subject, object$) is an element of the set of labels that dominates we say that

$$(subject, object) \in dominates$$

The dominance relation is defined as follows:

$$\forall (x1 \in subjects, x2 \in objects) : (x1, x2) \in dominates \Leftrightarrow \begin{aligned} &levels(x1) \geq levels(x2) \text{ and} \\ &category(x1) \supseteq category(x2) \end{aligned}$$

Hence the following are all true statements from the commercial label example above:

$$\begin{aligned} ((board, \{marketing, IT\}), (staff, \{marketing\})) &\in dominates \\ ((board, \{marketing, IT\}), (staff, \{production\})) &\notin dominates \\ ((staff, \{marketing, IT\}), (seniorExec, \{marketing\})) &\notin dominates \\ ((seniorExec, \{marketing, IT\}), (staff, \{marketing\})) &\in dominates \\ ((board, \{marketing, IT\}), (board, \{marketing\})) &\in dominates \\ ((board, \{IT\}), (board, \{marketing\})) &\notin dominates \end{aligned}$$

A Meta-data switching framework that would implement the formal definition has the following properties:

- There is a mathematical property about the data that is exploited and used for switching.
- Multi-Level security semantics is encoded into metadata and results in a single label.

In terms of our first definition - a meta-data switch needs to exploit a mathematical property; the Python Language directly supports ordinal numbers:

```
>>> ord('a')
>>> 97
>>>
>>> ord('b')
>>> 98
>>>
>>> 'b' > 'a'
>>> True
>>>
>>> 'a' > 'b'
>>> False
```

The dominance algorithm presented below directly exploits the ordinal number property of the Python programming language.

The second definition - Multi-Level security semantics is encoded into metadata and results in a single label. The steps are as follows:

First - we set-up the framework as a series of maps (Python dictionary structures)

```
>>> # Security labels
>>> # define a map of documents based data
>>> # classification
>>>
>>> classification = {'top-secret': 'e',
>>>                  'secret': 'd',
>>>                  'highly-protected': 'c',
>>>                  'unclassified': 'b',
>>>                  'public': 'a'}
>>>
>>> #define a map of documents based on
>>> #organisation Hierarchy
>>>
>>> level = {'board-level': 'c',
>>>         'executive': 'b',
>>>         'staff': 'a'}
```

```

>>>
>>> #define a map of documents based on hierarchy
>>> #of location
>>> location = {'global' : 'd',
>>>             'country' : 'c',
>>>             'state' : 'b',
>>>             'county' : 'a'}

```

Second – we will encode our security semantics as a series of characters that can be used in some kind of ordinal number test that will form the basis of our dominance function. We define an encoding function as follows and use this to encode the security semantics (the meaning) into the security labels.

```

>>> # define a simple encoding function
>>> def encode(aClassification, alevel, alocation):
>>>     clasn = classification[aClassification]
>>>     lev1 = level[alevel]
>>>     locn = location[alocation]
>>>     return clasn + lev1 + locn
>>>
>>> # define a simple decoding function
>>> # we will leave this to the reader
>>>
>>> # lets encode some values
>>>
>>> print encode('top-secret', 'board-level', 'global')
>>> ecd
>>>
>>> print encode('secret', 'staff', 'county')
>>> daa
>>>
>>> print encode('public', 'staff', 'state')
>>> aab

```

Note the encoding example ('top-secret', 'board-level', 'global') is translated to 'ecd'. Also note that 'ecd' is meaningless to the casual observer unless one understands ordinal number sorting is being utilized.

The ordinal number algorithm is a simple approach, but demonstrates the principles that we want to convey. We have developed and patented other industrial strength encoding algorithms where components of the algorithms can be kept secret with the encoder (Thus enabling further layers of obfuscation). The approach can also be combined with encryption if the transmission of security labels is required.

Third - A dominance function is introduced to allow or disallow the Subject access to the data / information Object:

```

>>> # define a simple ordinal dominance function
>>> def dom(aSubject, aObject):
>>>     count = 0
>>>     while 1:
>>>         # Object String and Subject string need to be
>>>         #the same size
>>>         if len(list(aObject)) <> len(list(aSubject)):
>>>             return False
>>>         #Check for end of Subject String and return
>>>         #True if found
>>>         elif count == len(list(aSubject)):
>>>             return True
>>>         #Use sort order test
>>>         else:
>>>             if list(aSubject)[count] < \
>>>                 list(aObject)[count]:
>>>                 return False
>>>             else:
>>>                 count += 1
>>>
>>> #some simple dominance tests
>>>
>>> dom('bbb','aaaa')
>>> False
>>>
>>> dom('bbb','aa')
>>> False
>>>
>>> dom('bbb','aaa')
>>> True
>>>
>>> dom('bbb','aab')
>>> True
>>>
>>> dom('bbb','aac')
>>> False
>>>
>>> # Now with the security semantics encoded we
>>> # see if the dominance function works
>>>
>>> # That is, does the subjects security label
>>> # dominate the information security label?
>>>
>>> # in the form dom(subject-label, information-
>>> # label)
>>>

```

```

>>> print dom(encode('top-secret', 'board-level', \
>>> 'global'), encode('secret', 'staff', 'county'))
>>> True
>>>
>>> print dom(encode('secret', 'staff', 'county'),\
>>> encode('top-secret', 'board-level', 'global'))
>>> False

```

Note the simple test in the dominance function to allow or disallow user role (Subject) access to the Object. In this case the ordinal “less than” operation between Subject and Object utilized, as a relation, would potentially fail the Subjects access to the Object if found to be true.

We didn’t need to decode the security label semantics to determine user role access; we just applied a very simple mathematical function.

Our alternative patented algorithms [6] do the same; they exploit some mathematical property to allow or disallow access once the semantics are encoded into the security labels.

IV. CONCLUSION

We have shown an approach where role-based security at an enterprise level can be further enhanced by implementing meta-data markings as security labels on User Roles and on Information (data).

Each security label post the encoding of the security semantics contains a mathematical property that can be used to switch (filter) information that users have requested access to.

In our example when comparing a Subject’s (user) label to the Object’s (information) label, the dominance function used an encoded mathematical property of ordinal numbers to allow or disallow individuals and groups to gain access to the information they requested.

The ordinal number comparison approach we used is simple and demonstrated the meta-data switching mechanism we wanted to articulate. However, we do not recommend using ordinal number properties. There are superior algorithms that we have patented.

The notion of the trusted system dominance function allows multilevel mandatory access controls for individuals and groups. The metadata-switching framework implementing the trusted system dominance provides an additional infrastructure control layer for the enterprise and has high performance characteristics that enable row level security (filtering) around any relational database.

A meta-data framework has two defining features:

- There is a mathematical property about the data that is exploited and used for switching.
- Multi-Level security semantics is encoded into the metadata and results in a single label.

ACKNOWLEDGMENT

We would like to thank Morris Levitzke for his proof reading of the paper.

REFERENCES

- [1] Amoroso, E. (1994), Fundamentals of Computer Security Technology, Prentice Hall, pages 70 – 78, USA.
- [2] Bell, David Elliott and La Padula, Leonard J. (1976) Secure Computer System: Unified Exposition and Multics Interpretation. MITRE Corporation, USA.
- [3] Clark, David D.; and Wilson, David R.; (1987) A Comparison of Commercial and Military Computer Security Policies; in Proceedings of the IEEE Symposium on Research in Security and Privacy (SP’87), May 1987, Oakland, CA; IEEE Press, pp. 184–193, USA
- [4] Denning, Dorothy E. A (May 1976) lattice model of secure information flow, Communications of the ACM archive, Volume 19 , Issue 5, pp: 236 – 243, USA.
- [5] Ferraiolo, D. and Kuhn, D. (1992). 15th National Computer Security Conference, Baltimore, Oct 13-16, 1992. pp, 554 – 563, USA.
- [6] International Patent Application PCT/AU2008/000702, Secure Keys Pty Limited, A Security Token and System and Method for Generating And Decoding The Security Token, 2008, Australia.