Space of Access Control Administration Models

Amir H. Chinaei
University of Calgary, CANADA
achinaei@ucalgary.ca

Ken Barker
University of Calgary, CANADA
kbarker@ucalgary.ca

Frank Wm. Tompa
University of Waterloo, CANADA
fwtompa@uwaterloo.ca

ABSTRACT

This paper proposes a novel formalism to classify access control administration models based on their expressivity in **a)** administrative decentralization and **b)** conflict resolution. The contribution is a taxonomy by which one can categorize a given access control admin model in the space of all models within two axes: axis of *decentralization degree of administration* and axis of *interpretation level of conflict resolution*. As guidelines to use the taxonomy, five degrees of decentralization and four levels of interpretation are developed. Finally, six well-known administration models, including the widely used System R, are compared by using this technique. As a bonus, the comparison intuitively critiques each model in terms of their administrative *functionality*, *performance*, and *security*.

Key Words: Access Control, (De)centralized Data Administration, Access Control Taxonomy, Conflict Resolution, Database Security.

1. Introduction

Access control is a critical aspect of any computer security mechanism currently used in modern computer environments. The overall security of a system can only be realized by guaranteeing a correct implementation of an organization's access control policies. Although there has been a large amount of research investigating new access control mechanisms and how these can be shown to be correct and consistent, these important contributions are very difficult to be quickly categorized even for expert users. Therefore, it is almost impossible to intuitively understand whether a new access control system is appropriate for an organization before going to its technical details. Hence, there is clearly a need to develop taxonomy techniques that are understandable by non-experts who need to use and manage access control admin models. Unfortunately, no intuitive access control taxonomy has been developed to provide this intuition in a well articulated way for non-expert security administrators, who are often called upon to determine who should be able to access the data they are responsible for managing.

To develop an intuitive usable taxonomy requires only the key aspects are articulated thereby avoiding the many subtle nuances associated with complex access control systems. Since the taxonomy is aimed at *admin* models, two aspects immediately present themselves: (1) the amount of decentralization present in administering the access and (2) the amount of conflicts that are tolerated by the admin model. The intuition behind the first feature can be thought of as the number of distinct individuals given

permission to update (i.e. manage) the system's access control data. The second feature provides an indication of how much tolerance the model has for the presence of conflicting rules that govern the access control policies. Thus, the former captures the decentralization complexity of an access control admin model while the latter captures its implementation complexity in the perspective of dealing with conflicts. Three intuitive observations are as follows. Minimizing either of these dimensions reduces the systems functionality while maximizing them increases its flexibility and utility. From performance point of view, maximizing the first feature while minimizing the second increases the system response time. From security point of view, minimizing the both features increases the safety property.

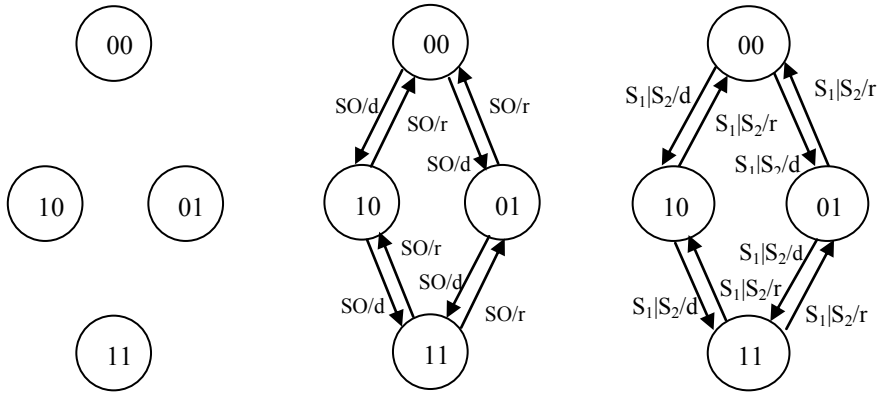
Unfortunately, no formal technique capable of comparing access control models in such a way has yet appeared in the literature. This paper contributes by developing an intuitive taxonomy by formally introducing the concepts of a *degree of decentralization* and a *level of interpretation* to ensure users of access control admin models have a solid understanding of their capabilities and complexity. Ultimately, this will be essential as we attempt to create systems that allow for personalized access in which the context awareness database system users will need to protect their own data.

We begin by briefly reviewing the conceptual technique of implementing an access control matrix in Section 1.1. In Section 2, we describe our taxonomy along two axes. Section 3 then demonstrates the taxonomy's utility by applying these axes to classify several well-known models. Finally, Section 4 summarizes our contributions and explores future directions.

1.1 Explicit vs. Effective Access Control Matrix

Access control data can be conceptually viewed as being represented by an access control matrix, where the rows represent subjects, the columns represent objects, and privileges are stored at the intersections [8]. The term *effective matrix* is used to represent effective privileges as a three-dimensional Boolean matrix M, indexed by subject, privilege (to execute a method), and object, in which no cell is null. The value of $M[s,m,o]$ is 1 if the corresponding subject s is privileged to execute method m on object o; otherwise $M[s,m,o]$ is 0 and the privilege is denied. Correspondingly, an *explicit matrix* implements the idea of condensing the effective matrix by storing explicit privileges only. The explicit matrix should be expanded to the effective one by using propagation and conflict resolution policies [3]. Section 2.2 uses our *interpretation level* to classify such policies.

Proper management and updating of the explicit matrix can be accomplished in various ways, with respect to decentralization. For example, in some systems only a designated subject type (often called the *security officer*, which can be a group too) can update the matrix while other systems allow several subjects such privileges. In some other systems, a hierarchy of administrators manages the access control data. Furthermore, there are applications in which the object creator is responsible for managing the access to that object. Note that there is no single "correct" paradigm since different environments demand various protocols but it is important to recognize the *spectrum of access control administration* so users understand their implications. At one end of the spectrum, access control can be absolutely *autocratic*: a powerful administrator exists in the system dictating which subjects have access to which objects; at the other extreme, it can be completely *self-governing*, whereby no central administrator exists in the system, but users fully manage their own data. A given access control admin model can fit anywhere into the spectrum. Mandatory access control models reflect the former extreme while discretionary access control



(a) *D*(0) **(b) *D*(1)** **(c) *D*(*n*)**

Figure 1. Decentralization degree and transition labels.

models approximate the latter. Section 2.1 exploits our *decentralization degree* to formalize this aspect of access control.

2. Formalism

Recall from Section 1.1 that there is a spectrum of access control admin models that covers all models from the autocratic end (very centralized) to the self-governing end (very decentralized). Also recall that regardless of how decentralized the explicit matrix is updated, there are a variety of policies to derive an effective matrix. In other words, every single point along the first axis can be interpreted to multiple points along the second axis. Sections 2.1 and 2.2 formalize these aspects by two axes, respectively. Then, in Section 2.3, the axes coalesce to define the space of access control administration models.

2.1 Decentralization Degree of Administration

Decentralization degree of an access control admin model, denoted by $D(m)$ where m is a non-negative number, specifies how many distinct subjects (subject types) are privileged to update the explicit access control matrix. For instance, $D(0)$ represents a system in which the explicit matrix cannot be updated once initialized; and, $D(1)$ represents an autocratic system in which only one subject (or one group of subjects), which is often called the *security officer*, can update the matrix.

Decentralization degree represents a coefficient of the number of transition labels for a network of access control states, similar to a finite state automaton. For instance, assume that a given system contains one object, one privilege, and two subjects, called S_1 and S_2. The explicit matrix of such a system consists of four states, namely 00, 10, 01, and 11, in which the left (or right) digit represents the accessibility of subject S_1 (or subject S_2) to the object; 0 means no accessible and 1 means accessible. (Note that, for simplicity, we avoid null values.) Hence, these states represent neither S_1 nor S_2, only S_1, only S_2, and both S_1 and S_2 have the privilege on the object, respectively. Figure 1 illustrates three different decentralization degrees for such a system, as examples. Figure 1(a) represents $D(0)$ in which the explicit matrix is not updated at all in the system life cycle. In particular, the initial state of the system is one of 00, 10, 01, and 11, and there is no transition from the initial state to any other state. $D(0)$ demonstrates *one extreme of the spectrum* of access control models, in which the administration is too centralized (no change at all at run time). Figure 1(b) represents $D(1)$ in which one subject SO (the security officer) administers the

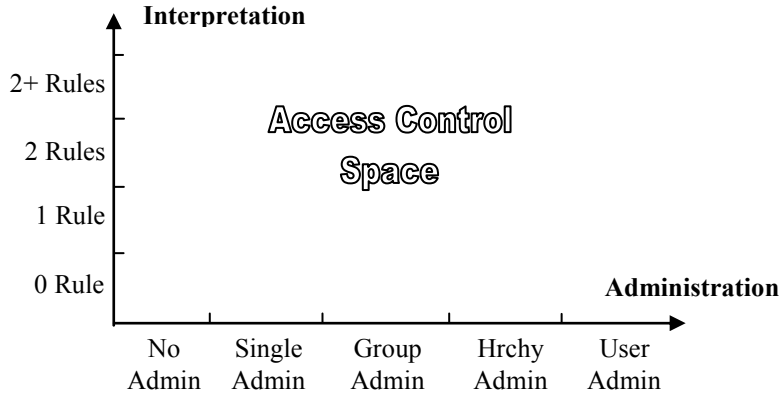


Figure 2. Space of access control administration models.

explicit access matrix. The transition labels are SO/d or SO/r, which means the security officer delegates or revokes the privilege to/from any of the subjects, respectively. Figure 1(c) demonstrates *another extreme of the spectrum*, *D*(n), an anarchistic case in which all subjects (here S_1 and S_2) can administer the explicit matrix without any constraint. The transition labels are S_1/d, S_1/r, S_2/d, or S_2/r, which means both subjects can delegate (and revoke) the privilege to (from) one another. (This is anarchistic since S_1 and S_2 may alternatively inverse a cell of the explicit access control matrix, repeatedly in a loop.)

Note: It is important to note that this paper is not addressing the state-reachability problem; instead, the focus is on how decentralized the transitions are for a typical access control admin model and how diverse a specific state can be interpreted in that same model.

2.2 Interpretation Level of Conflict Resolution

Interpretation level of an access control admin model, denoted by *I*(*n*) where *n* is a non-negative number, determines how many different interpretations can be provided by the conflict resolution component of the model. For instance, *I*(0) represents a system in which no conflict resolution component exists (in such a system, a conflict is treated as an error); and, *I*(1) represents a model in which there is only one conflict resolution policy, such as *positive-takes-precedence*, and that policy is hard wired to the system.

As an example, assume that a given system contains one object, one privilege, and two subjects, called S_1 and S_2. Moreover, assume that S_1 is a group of which S_2 is a member. Therefore, the effective privilege of subject S_2 in both states 01 and 10 of Figure 1 is subject to conflict resolution—because the member and the group have opposite access to the object. There are several strategies, beyond the context of this paper, to resolve such conflicts. (Interested readers should consult [3].) The higher the interpretation level is, the more variety of conflict resolution strategies it supports.

In Section 2.3, we explain how the decentralization degree and the interpretation level serve as x-axis and y-axis, respectively, to define the administrative space of access control models.

2.3 Space of Access Control Administration Models

This section consolidates the decentralization degree of access control administration and the interpretation level of conflict resolution to introduce the space of access control administration models. Decentralization degree of an access control admin model determines how decentralized the *explicit* access control matrix can be administered. Interpretation

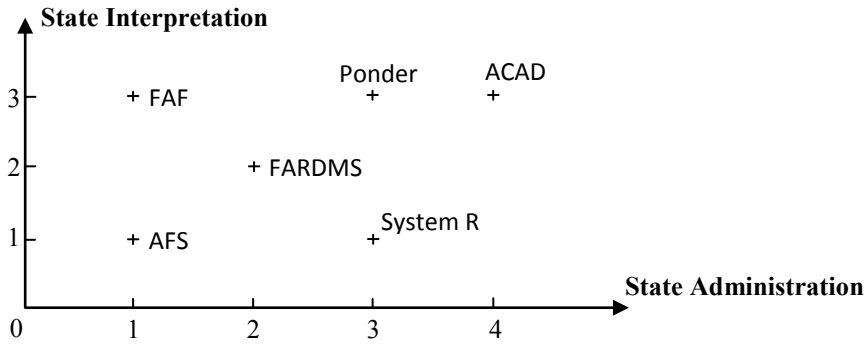


Figure 3. Comparison of six models in access control space.

level of a conflict resolution model determines how diverse each state of the explicit access control matrix can be interpreted to an *effective* matrix. As an analogy, decentralization degree represents a coefficient of number of transition labels for a network of access control states, similar to a finite state automaton, whereas conflict resolution policies provide different *interpretations* for each state. In a flexible access control admin model all reachable states as well as their interpretation conform to the access control policy chosen by the enterprise.

Figure 2 represents such a space, in which one axis (called *State Administration*) maps the degree of decentralization and the other (called *State Interpretation*) maps the level of interpretation. For the sake of comparison, we partition the Administration axis to represent five *classes* of models with respect to the amount of administrative decentralization, namely *no admin*, *single admin*, *group admin*, *hierarchical admin*, and *user admin*. (However, some readers may prefer to come up with a different partitioning.) Systems in which there are no metadata updates and each component authority is fixed in the life cycle are from the no admin class. A system, such as 4Dl-IRIS OS, in which only one user is allowed to update some data, is from the single admin class. Similarly, UNIX, in which a group of users may take the role of super-user, is from the group admin class. Role-based access control models, in which roles often map the organizational hierarchy, are from the hierarchical admin class. Another example of hierarchical administration is the security model of System R [5]. Finally, user-managed access control models (introduced in [2]), in which each user potentially can administer various parts of the system, are from the user admin class. For simplicity of discussion, we identify the administration classes with numbers 0, 1, 2, 3, and 4, respectively. It is important to notice that the classes are partially ordered, and the higher the number is, the more flexible class it represents. Hence, models in Class 0 (no admin) can be described by models in Class 1 (single admin), and so on.

Similarly, we partition the State Interpretation axis to represent four *levels* of models with respect to the flexibility of the conflict resolution component, namely *0 rule*, *1 rule*, *2 rules*, and *2+ rules*. (Again, some readers may prefer to come up with a different partitioning.) Level *0 rule* represents access control admin models in which conflicts are not possible or allowed; an error is raised in the latter case. Level *1 rule* represents access control admin models in which conflicts are resolved by one rule only, for instance *negative-takes-precedence*. Level *2 rules* represents models in which conflicts are resolved by two rules, for instance *the-most-specific-takes-precedence* and if there is still a conflict then *positive-takes-precedence*. Level *2+ rules* represents models in which the conflict resolution component is not hard wired to the system and can be replaced by any conflict resolution strategy. Similar to the administration classes, we identify the

interpretation levels with numbers 0, 1, 2, and 3, respectively; and, the higher the level is, the more variety of interpretations it represents. Hence, models with Level 0 (no conflict resolution) can be described by models with Level 1 (resolving conflicts by 1 rule), and so on.

We suggest to represent the expressivity of each model by a pair of <class, level>, where class identifies the model administrative capabilities and level identifies its support of variety of conflict resolution strategies. Therefore, the space of access control admin models provides a visualized mechanism to compare existing models, and leads to a better understanding of their functionalities as well as highlighting their overlaps and differences.

3. Case Study: Comparison of Models

In this section, by applying the metrics introduced in Section 2, we classify several noteworthy access control admin models, so called AFS, System R, FARDMS, FAF, Ponder, and ACAD. AFS [6] is the security model for the Andrew File System. System R [5] is the first relational database management system. It is also among the firsts systems to permit users to share and control their data in a multi-user environment. FARDMS [1], Flexible Authorization model for Relational Databases Management Systems, extends the System R model by supporting access control exceptions and strong enforcement. FAF [7] is a specification language to support various access control policies in a system. Ponder [4] is a declarative policy specification language for management and security of distributed network systems. ACAD [2] is an access control model, which supports variety of user-managed administration paradigms.

Figure 3 illustrates the position of AFS, FARDMS, FAF, System R, Ponder, and ACAD within the access control space. In terms of administrative capabilities, AFS and FAF are in the administrative Class 1 since they allow only one user to be the security administrator. However, it is clear that both models can be extended to support a group of administrators, with equal capabilities, and therefore be in Class 2. FARDMS is in the administrative class 2 since it currently supports a group of privileged subjects to take administrative capabilities. System R and Ponder, in Class 3, provide a more general administrative model—with respect to previous models—since they support a hierarchical administration, for instance appropriate for distribute computer networks. However, it is obvious that hierarchical admin models cannot express graph-based admin models such as the user managed access control model supported by ACAD. Therefore, ACAD, in Class 4, is the most flexible model in terms of decentralization with respect to other existing models.

In terms of interpretation variety, AFS and System R are both in Level 1 since they support a single rule of *negative-takes-precedence*. FARDMS, in terms of interpretation variety, is richer than AFS and System R since it supports the combination of two rules, the *most-specific-takes-precedenc*e and *negative-takes-precedence*, and is in Level 2 then. (FARDMS also supports the notion of *strong* and *weak* authorizations [1], which is beyond the context of this paper.) Moreover, the conflict resolution component in AFS, System R, and FARDMS is hardwired to the rest of model, which causes support of different strategies difficult. However, FAF, Ponder, and ACAD are all in Level 3, which reflects that they are independent from the conflict resolution component. FAF and Ponder explicitly support any combination of three rules of *the-most-specific-takes-precedenc*e, *negative-takes-precedence*, and *positive-takes-precedence*, which is equivalent to two strategy instances [3]. ACAD explicitly supports four rules of *locality*, *majority*, *default* and *preferred*

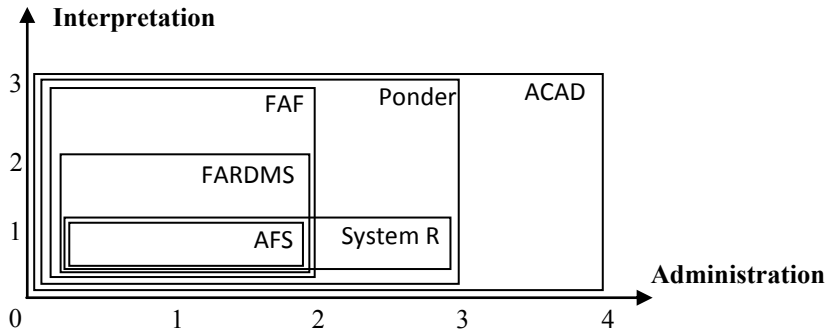


Figure 4. Expressivity of six models in access control space.

authorizations, which covers 48 conflict resolution strategies including the ones supported by above models [2].

In summary, the expressivity of AFS, FARDMS, FAF, System R, Ponder, and ACAD can be represented by <1,1>, <2,2>, <1,3>, <3,1>, <3,3>, and <4,3>, respectively. Considering the fact that AFS and FAF are simply extensible to <2,1> and <2,3>, respectively, one can easily conclude that, in terms of administrative capabilities, these models obey the following rules

1- AFS < FARDM < FAF < Ponder < ACAD
2- AFS < System R < Ponder < ACAD

in which "<" means "can be captured by". This is illustrated by Venn diagram in Figure 4.

As discussed in Section 1, such taxonomy intuitively provides potential (non-expert) security officers with some justifying information about *functionality*, *performance*, and *security* of the access control admin models of their interest. For instance, above rules intuitively state that the access control component of AFS is less functional (in terms of decentralization), more efficient (in terms of performance), and probably safer (in terms of security) than that of Ponder, without going to technical details of each model. Yet, some users may still want to investigate each model more after gaining such insights.

4. Summary and Future Work

This paper introduced the degree of decentralization and the level of interpretation for access control admin models. The decentralization degree of an access control admin model determines how decentralized an explicit access control matrix can be administered. The interpretation level of a conflict resolution component determines how diverse an explicit matrix can be transformed to an effective one. This paper also brought together these two aspects to define the space of all access control models. As a case study, several models were classified using the proposed metrics. Several parties can benefit from our metrics and taxonomy: security system developers can adjust their products to meet their user requirements; system buyers can evaluate the existing systems and compare their administrative functionalities before purchase; researchers can verify security models in terms of their expressivity in decentralization and interpretation as well as their vulnerability to attacks, such as information flow.

We plan to extend this work by investigating other partial orders representing each dimension of the space. This may lead us to two important directions as follows.

Define a formal technique to verify decentralized access control administration models in terms of "the degree of decentralization". Decentralization may increase anarchy, and centralization may cause an administration bottleneck. In other words, decentralization, e.g. in information sharing systems that fit into DAC models, is a special type of optimization problem in which the degree

of decentralization needs to be maximized while keeping the anarchy below a specific amount. Similarly, centralization, e.g. in governments that often fit into MAC models, is an optimization problem in which the degree of centralization needs to be maximized while keeping the administration load below a specific level.

Define formal metrics to measure the restrictedness degree. Conflict resolution policies together with propagation policies raise an interesting question: how restricted is the combined system overall? Intuitively, this question addresses the ratio of positive and negative authorizations in the effective access control matrix. Developing such dimensions—to measure the degree of restrictedness of a system—could result in two immediate profits: first, understanding if a given access control admin model approximates *closed policy* systems or *open policy* ones, which consequently has several advantages, including choosing an efficient data structure; second, such metrics could help in verifying some properties of access control models such as the data availability and safety properties.

Acknowledgements

We gratefully acknowledge financial support from the Universities of Calgary and Waterloo.

References

1. Bertino, E., Jajodia, S., and Samarati, P. A flexible authorization mechanism for relational data management systems. ACM Transactions on Information Systems, 17, 2 (April 1999), 101-140.

2. Chinaei, A. H. "Access Control Administration with Adjustable Decentralization," PhD. thesis, University of Waterloo, Waterloo, Canada, 2007.

3. Chinaei, A.H., Chinaei, H.R., and Tompa, F. Wm. A unified conflict resolution algorithm. In *Proceedings of the 2nd VLDB Workshop on Secure Data Management*, Vienna, austria, September 2007. Springer-Verlag Berlin Heidelberg, 1-17.

4. Damianou, N., Dulay, N., Lupu, E., and Sloman, M. The Ponder policy specification language. In Proceedings of the Policy Workshop on Distributed Systems and Networks. Bristol, UK, January 2001. Springer-Verlag LNCS 1995, 18-39.

5. Griffith, P.P., and Wade, B.W. 1976. An authorization mechanism for a relational database system, ACM Transactions on Database Systems, 1,3 (1976), 242-255.

6. Howard, J.H., Kazar, M.L., Meness, S.G., Nicholas, D.A., Satyanarayanan, M., Sidebotham, R.N., and West, M.J. Scale and performance in a distributed file system. ACM Transactions on Computer Systems, 6,1 (February 1988), 51-81.

7. Jajodia, S., Samarati, P., Sapino, M.L., and Subrahmanian, V.S. Flexible support for multiple access control policies. ACM Transactions on Database Systems (TODS), 26, 2 (June 2001), 214 - 260.

8. Lampson, B.W. Protection. In Proceedings of the 5th Annual Princeton Conference on Information Sciences and Systems, Princeton, New Jersey, USA, March 1971, 437-443.