Combining Mandatory and Attribute-based Access Control

Lawrence Kerr  
University of Idaho  
lawrence.kerr@vandals.uidaho.edu

Jim Alves-Foss  
University of Idaho  
jimaf@uidaho.edu

Abstract

Attribute based access control (ABAC) offers a great deal of flexibility over more traditional forms of access control in that it relies less on user identity or role but on various attributes of a subject or object. In many instances where a traditional access control approach is taken, such as mandatory access control (MAC) environments, more information beyond a classification is desirable to make a more flexible access control determination. We propose an ABAC model that retains the nature of a strictly MAC approach, while enriching access control decisions with a number of other security attributes by leveraging the concept that classification, clearance, or any other security property of a subject or object is simply an attribute. The model description is followed by an example instance based on current DoD guidelines.

1. Introduction

A variety of attribute-based access control (ABAC) approaches have been proposed recently. These range from specific applications to generalized models demonstrating ABAC’s ability to model other traditional access control approaches such as mandatory (MAC), discretionary (DAC), and role-based access control (RBAC) models.

ABAC shows promise, largely in its flexibility. Government has begun to realize the benefits of such an approach to access control [1]. In such an environment, MAC needs collide with a broader range of concerns. Now, as opposed to simply a classification and compartments, a data object might be subject to a number of different properties describing its access control posture [2]. These properties might include dissemination or export controls, expiration dates, or other special program markings. ABAC can certainly handle these extended attributes, but what of maintaining the MAC properties of the system?

This paper presents a combined MAC and ABAC model derived by treating the classification, clearance, and compartments of a traditional MAC model as attributes within an ABAC model. By distinguishing classification, clearance and compartments mandatory attributes for each subject or object in the system, we generate a model that accommodates both traditional MAC aspects and maintains more flexible ABAC properties.

The remainder of this paper is organized as follows. Section 2 explores background research on MAC and ABAC models. Section 3 presents an ABAC model that incorporates traditional MAC approaches, while Section 4 offers an example implementation of the model based on existing real-world policies. Section 5 includes future work and concluding remarks.

2. Background

While MAC is a fairly well established model, ABAC is still in its early stages, and lacks a consensus definition. A great deal of literature exists regarding ABAC in a variety of environments that demonstrates its flexibility and applicability to many situations.

2.1. Mandatory access control

MAC is an access control method with a relatively long history [3]. It is based on the predetermined imposing of classifications on objects (data), with like clearances assigned to subjects (users). Access control decisions are made based on the relationship between object classification and subject clearance. The classification (and clearance) is a totally ordered set. Often times, a set of compartments is associated with each subject and object. Each compartment represents some “need-to-know” concept and supplements the classification or clearance. Taken together, the totally ordered set of classifications (or clearances) and all possible compartments form a lattice.

The Bell La Padula model is the classic MAC example which provides a model of multi-level security [3]. In this model, confidentiality of information is the main focus. Three main properties are observed. First, the simple security property
requires that a subject is only allowed to read objects at or below his or her clearance with a subset of his or her compartments. This is known as the dominates relationship.

The second property of the Bell La Padula model is the \*\textit{property}. This property ensures that no higher level information is written to a lower level object. In its basic form, this equates to a subject only writing to any object with a classification and compartments that dominates that of the subject. In a strict \*\textit{property}, this constraint is tightened, with the subject only able to write to objects with a clearance and compartments matching that of the object.

Finally, the tranquility property ensures that subjects and objects do not change in their respective classifications or clearances while performing operations.

### 2.2. Attribute-based access control

A number of different approaches to ABAC have been proposed in the literature, there does not yet seem to be a clear consensus as to an exact definition or model of ABAC. Common to much of the work however, is the advantage of ABAC in context-aware or pervasive computing, where the identity of a service consumer is not necessarily needed or perhaps even known.

Wang et al. [4] discuss an approach to ABAC where attributes and services are represented as sets. Using a logic programming approach, three basic predicates are used, supplemented as needed with domain-specific predicates. These basic predicates combine to form larger policies that define what sets of attributes are necessary to access sets of resources. The authors provide an example digital library system providing a number of services. These might include a browsing service or printing service. The browsing service provides capabilities specific to various media types: browse tables of contents, abstracts, or content. Services are represented by their paths: \{dls, br, brToC\}, \{dls,br, brContent\}, etc. Users have attributes such as membership level, method of payment, printing preferences, and so on. Predicates are formed using these resource sets with specific user attribute conditions. For example, a basic member is only granted access to the \{dls, br, brAbstract\} set, whereas a senior member can access all capabilities provided by the browsing service. In this example, the system does not necessarily need the identity of the requesting user, only the user attribute designating membership level to grant or deny access.

For an attribute-based messaging system, Bobba et al. [5] construct policies from attribute name:value pairs. Access control policies here consist of a condition that when satisfied, grant a user access to different message recipients or groups. Values can be enumerated or discrete. A condition is a check on values associated with one or more attributes in conjunctive normal form (CNF). For this system, checks on values include \(=\), \(<\), \(>\), \(\leq\), and \(\geq\). These conditions form the policy that in this case governs if a user with a particular set of attributes can send a message based on the attributes of the recipient.

The system proposed by Bobba et al. [5] is not without issues. As the authors point out, large sets of conditions are possible if each name:value pair is considered its own policy. To avoid this, the use of generic forms is encouraged. In a course instructor example, writing a rule such that a user holding the instructor attribute of the given course is allowed an action is more generic than writing a rule for each course-instructor pairing.

One other interesting observation of Bobba et al. is the presence of sensitive attributes [5]. In this case, all attributes considered in the rule should be assumed to be sensitive. This concept is similar to the work of Frikken et al. [6], who seek to hide policies from consumers, while at the same time hiding credentials from providers in an ABAC environment. This maintains the confidentiality of the consumer’s attributes which may contain sensitive information, as well as the provider’s policies to prevent a malicious party from knowing which attributes require forging to gain access.

Cirio et al. [7] extend a role based access control (RBAC) model with ABAC. They present the difficulties with a RBAC system such as the static nature of role assignments. ABAC supplements RBAC here, adding flexibility through use of attributes for role determination as opposed to user identity. Attributes are associated with both users and resources, allowing dynamic specification of privileges for resources and association of users with privileges. They provide a proof-of-concept system leveraging their Web Ontology Language (OWL) based Descriptor Language (DL) to describe RBAC and ABAC aspects of the system.

Jin et al. [8] state the necessity of more formally defining ABAC, while providing a model for ABAC that is capable of expressing other, more traditional models such as mandatory, discretionary, or role based access control. Under this model, each attribute is a function with a specific range that returns a value or set of values for some entity. Entities include users, subjects acting on behalf of users, and objects representing the resources available in the system. Each entity is associated with a finite set of attribute functions which return
properties of the associated entity. Policies are constructed using constraints on the values of these attribute functions.

Once the basic entities and attribute sets are defined, four configuration points are defined: 1) authorization policies 2) subject attribute constraints, 3) object attribute creation time constraints, and 4) object attribute modification constraints. Authorization policies return true or false as access is granted or denied. Using this framework, the authors are able to create ABAC policies that adhere to DAC, MAC, and RBAC policies.

In work combining RBAC and ABAC, Kuhn et al. [9] point to a number of issues in selecting one approach over the other. A purely ABAC approach with n attributes may require as many as 2^n rules, while a worst case RBAC system requires the same number of rules to cover all possibilities. ABAC is easily configured and presents the flexibility necessary to handle dynamic environments. This flexibility comes at a cost: changing or analyzing permissions can become a complex task as the number of attributes increases. RBAC presents easier administration, with initial costs of establishing the role hierarchy.

Kuhn et al. [9] present a combined role-centric model RBAC-A, where attributes are used to supplement and further constrain an RBAC model. Roles and attributes are distinguished from one another based on whether they are static or dynamic. Attributes that are static, or reasonably static, are used as the basis for roles. These include things such as office location, position, or nationality. They are not likely to change frequently if at all. More dynamic attributes such as time of day are leveraged in the ABAC portion of the combined model. Using static roles and dynamic attributes together can significantly cut down on the number of possible roles and rules. An example system with 4 static attributes and 6 dynamic results in at most 2^4 roles and 2^6 rules, whereas a strictly RBAC or ABAC approach results in as many as 2^10 roles or rules, respectively. While this combination still presents 2^10 possible combinations, the total number of roles and attributes to track is much less.

3. MAC and ABAC Model

The MAC influenced ABAC model detailed below is intended for use in environments where a MAC policy is desirable, but further factors beyond the clearance of the user is necessary to make appropriate access decisions. The model describes the general case for combining MAC policies into ABAC. A MAC/ABAC combined model M is described as:

\[ M = \{S, O, A_s, A_o, A_e, C, P, R\} \]

where S is the set of all subjects; O is the set of all objects; \(A_s, A_o, A_e\) are sets of all possible attributes for subjects, objects, and the environment, respectively; C is a set of constraints on attributes; P is the set of possible operations a subject performs on objects; and R is the set of all attribute based access decision rules.

Attributes here are defined as any security relevant property associated with a subject, environment, or object. Following from the Jin et al. [8] approach to ABAC, the classification of objects or clearance of subjects, as well as compartments are treated as attributes. All attributes used will be either atomic or set based, with each set consisting of a number of atomic attributes. An atomic value might be a classification of a data item as it is typically a single string value, whereas the compartments associated with a particular object are a set value where any number of valid atomic valued compartments could be set members. Operators used will be specific to the type of attribute they operate on. Atomic operators include <, =, and ≤, while set operators include \(\subset, \subseteq, =, \text{ and } \notin\).

This model features not only subject and object attributes, but also environment attributes. For our purposes, subjects represent users of the system. Subject attributes are those attributes which describe some aspect of the subjects accessing resources. Objects in this case represent the resources or data the subjects attempt to access. Environment attributes include attributes which do not apply to specific subjects or objects, but rather the context in which access decisions are made.

To retain MAC properties in this ABAC model, mandatory attributes are specified for subjects and objects. These attributes are explicitly mandatory in all user requests, so any request that does not include all mandatory attributes must be rejected. These include subject clearance, object classification, and compartments. These attributes allow retaining the dominates relationship from the MAC model, as well as the simple security and *-properties.

Mandatory subject attributes include the clearance and compartments of a subject, while attributes include such things as nationality, pay grade, position, or employing agency.

Objects have mandatory attributes including classification, compartments and present, but may cover a number of other items. Other attributes, such as For Official Use Only (FOUO) or Sensitive But
Unclassified (SBU) are included to describe otherwise unclassified information that still requires some level of protection. Other dissemination controls such as to whom a particular object is releasable to, or whether the information comes from an open or protected foreign source are desirable. Date classified, original classifier, and duration of classification may also appear as object attributes.

While user and object attributes will tend more towards static attributes, environmental attributes are more dynamic. These might include source address for access over networks, time of day, or perhaps the number of accesses already requested by the user within a particular time window.

A set of constraints on attributes is necessary, describing the allowed values or ranges, relationships between different attributes, and how attributes are applied as new objects are created. A constraint on attribute values might be used to limit an attribute to a specific range or set of values. An example is the classification of an object. The classification is expected to be a member of a finite set of atomic values, whereas the compartments could be any subset of the set of all compartments:

\[
\text{class } \in \{U, C, S, TS\} \\
\text{comps } \subseteq \{\text{red, blue, green}\}
\]

Further constraints may be among related attributes. In cases where one attribute determines the value or existence of another:

\[
\text{if attrib}_1 = \text{value then attrib}_2 = \text{value}
\]

This type of constraint is used in the cases such as unclassified information that is not compartmentalized. In this case, if the classification is U, then the compartments attribute is assigned the value of the empty set.

Each attribute rule returns a boolean value for a subject’s operation on a specific object given two attributes of interest, or an attribute and a literal. The attributes are compared using a supported operator. Jin et al. [8] acknowledge the possibility to generate a third “don’t know” value from a rule. This is unacceptable, however, in our MAC environment, so rules will be limited to only returning true or false.

For example, a rule to determine if a classification has expired would include a comparison of the current date environment attribute (returned by the e.now method) and the classification expiration date object attribute (returned by the o.expiration property):

\[
e.\text{now} < o.\text{declass}
\]

Access control decisions are made by a collection of all attribute rules for a specific operation as a single statement in CNF form. For a particular permission, each applicable attribute rule must be satisfied. For example, a policy describing a strict *-property whereby a subject can only write to an object at the same level:

\[
\text{write} = (o.\text{class} = s.\text{clear}) \land (o.\text{comps} \subseteq s.\text{comps}) \land (s.\text{comps} \subseteq o.\text{comps})
\]

where subject s has requested a write to object o, o.class returns the classification of o, o.comps returns the compartments of o, s.clear returns the classification of s, and s.comps returns the compartments of s.

One critical issue here is that not every object will have every, or perhaps even most, of the attributes. When access control decisions are made, only those object attributes that are present will be used to determine which rules are to be evaluated. Missing object attributes will not be used as they lack values of the object and will not have an impact on the decision.

By evaluating only specific sets of attributes as access control decisions are made, a greater degree of flexibility is allowed over other approaches [8] in which a single rule for each allowed action is created. Dynamically assessing attributes allows for easier adding of attributes over time, as only the subjects and objects affected by the new attributes need be updated, and appropriate rules can be integrated. With a single rule that encompasses all attributes, each time a new attribute is added, it must be added to all subjects and objects, and each rule must be updated.

4. Example instance

As a demonstration of the above model, we implement a subset of the United States Department of Defense classified information marking policy as defined in manual 5200.01, Volume 2 [2]. This document outlines 23 possible object attributes in use beyond classification and compartments. In addition to these object attributes, this policy describes three implications on the use of or relations between attributes (e.g. the JOINT label indicates information jointly owned by the US as well as at least one other nation or international organization, implies releasability to the joint owners without requiring the REL label) and 16 constraints on the usage (e.g. information that bears a NOFORN label cannot be released to foreign nationals, and therefore cannot
also bear the REL label). These labels and procedures within the policy imply twelve subject attributes. Considering at least one rule for each of the 23 object attribute labels, eight rely on the user nationality attribute, while other subject attributes may only relate to single object attributes. The subset of these considered here includes thirteen object attributes, and seven subject attributes. A single environment attribute, date, is used.

This example assumes a dynamic environment where subjects interact with documents. Objects in this case are the paragraphs that make up each document. Each paragraph is associated with some number of attributes. Paragraphs within a document are not guaranteed to bear identical attribute sets, as different subjects create modifications over time. At all times, the simple security property, the \( * \)-property, and tranquility must be maintained.

Each subject is assigned a set of attributes as well. This set of attributes establishes the "need-to-know" of the subject. A subject with a secret clearance belonging to the red and blue compartments, for example, has read access objects with secret classification belonging to any combination of red and blue compartments. If this subject happened to be a German citizen, however, access to objects bearing an attribute restricting them to only US citizens would be denied. Adding the nationality attribute to the access control decision in this instance limits permitted operations.

4.1. Object attributes

To maintain MAC properties, we first include the atomic mandatory attribute classification as a totally ordered set, where unclassified (U) is the lowest level, dominated by confidential (C), the secret (S), with top secret (TS) as the highest level representing the most sensitive data. In addition to the classification, compartments are necessary. The compartments attribute is represented by a set of compartments that relate to the object. There is no ordering among compartments; access control decisions consider set membership only. These attributes, as well as all others, are predetermined for each subject and object.

Other object attributes are necessary to adequately describe the "need-to-know" for a particular object. From DoD 5200.01, Volume 2 [2] we consider the following:

- Foreign government information (FGI) marks information as sourced from foreign governments or organizations. This attribute consists of a set value, with members reflecting which countries or organizations provided the information in the object.
- Joint (JOIN) information, like FGI information, comes from foreign governments or organizations except in this case, the US is one of the source countries. As the US is assumed an information source, it is not required to be listed in the set.
- Releasability of US information to foreign countries or organizations is noted with the RELTO attribute. This is a set value that contains the countries the object is releasable to.
- No foreign (NOFORN) information is not releasable to any non-US entity.
- For official use only (FOUO) or sensitive but unclassified (SBU) apply to unclassified information that still requires some measure of protection. These are atomic values.
- Special Access Programs (SAR) attribute represents a set of program names.
- Atomic Energy Program information presents a number of possible attributes: restricted data (RD), formerly restricted data (FRD), and unclassified controlled nuclear information (UCNI). These are atomic values.
- Declassification date is used for automatic declassification, and is typically 25 years from the date of the object's origin.

Objects are represented as a set of attributes, for example:

\[
o = \{\text{class}=S, \text{comps}=&\{\text{red, blue}\}, \\
\text{JOIN}=&\{\text{GBR, DEU}\}, \text{declass}=20160101\}\]

describes object \( o \) with a secret classification, red and blue compartments, a declassification date of January 1, 2016, and jointly sourced information from the US, Great Britain, and Germany.

4.2. Subject attributes

In accordance with the MAC aspects of this example, each subject is assigned a clearance identical to the object classification. Additionally, each subject is associated with some number of compartments.

Aside from mandatory attributes, each subject bears a nationality attribute representing the country of current citizenship. This is typically a single atomic value, but there are cases (dual citizenship, for example) where this must be a set value. To accommodate all cases this will be a set value,
containing all countries of which the subject is a citizen.

Remaining subject attributes include a set value for Special Access Programs. This set includes all programs to which the subject has been granted access. Additionally, an attribute for each of the Atomic Energy Program object attributes is necessary: RD, FRD, and UNCI.

Similar to objects, subjects are represented as sets of attributes:

\[ s = \{ \text{clear}=\text{TS}, \text{comps}=\{\text{blue}, \text{green}\}, \text{citizen}=\{\text{US}\}, \text{FRD}=\text{true}, \text{UNCI}=\text{true} \} \]

### 4.3. Attribute constraints

Attribute constraints serve to ensure attributes possess valid values and do not conflict with other attributes, particularly at object creation time. This work assumes a strict *-property, where subjects only create objects bearing a classification and compartment set matching the creator’s clearance and compartment set.

One constraint here occurs when the presence of one attribute implies the presence of another. Such is the case with the JOINT attribute. When the JOINT attribute appears, it implies that the RELTO attribute is also present with the same set of countries or organizations:

\[ \text{if } o.\text{exists(JOINT)} \text{ then RELTO } = \text{JOINT} \]

where the exists method returns true when the object \( o \) has a valid value for the JOINT attribute.

As discussed above, unclassified information is not compartmentalized, and thus has an empty set for the compartments attribute value:

\[ \text{if } o.\text{class} = \text{U} \text{ then } o.\text{comps} = \emptyset \]

Many attributes can occur only at specific classification levels:

\[ \text{if } o.\text{class} \notin \{\text{C, S, TS}\} \text{ then RD } = \text{false} \]

\[ \text{if } o.\text{class} \notin \{\text{C, S, TS}\} \text{ then FRD } = \text{false} \]

\[ \text{if } o.\text{class} \in \{\text{C, S, TS}\} \text{ then UNCI } = \text{false} \]

Other attributes are counter-indicated. For example, a NOFORN attribute conflicts with JOINT, FGI, or RELTO attributes:

\[ \text{if } o.\text{exists(JOINT)} \text{ then NOFORN } = \text{false} \]

\[ \text{if } o.\text{exists(FGI)} \text{ then NOFORN } = \text{false} \]

\[ \text{if } o.\text{exists(RELTO)} \text{ then NOFORN } = \text{false} \]

### 4.4. Allowed operations and rules

In this example, as subjects interact with various objects within documents, two basic operations are identified: read and write. While other operations may occur, they will be extensions of these operations so those rules will not be listed here. Each operation has a number of associated rules, with each rule based on different subject, object, and environment attributes. For the purposes of this work, we adopt a strict *-property, where a subject can only write to an object bearing a classification and set of compartments equivalent to the clearance and compartments of the subject.

For each read access rule below, subject \( s \) seeks to read the contents of object \( o \) in an environment \( e \). Each rule results in a boolean return value representing whether the read access is allowed (true) or rejected (false).

The MAC read operation is composed of two individual rules:

\[ \text{classRead } = (o.\text{class} \leq s.\text{clear}) \]

\[ \text{compRead } = (o.\text{comps} \subseteq s.\text{comps}) \]

If these two rules are satisfied, additional access control decisions are made based on the attributes of the object. Only attributes the object possesses are used.

\[ \text{fgiRead } = (s.\text{citizen} \subseteq o.\text{FGI}) \lor (o.\text{citizen} = \{\text{US}\}) \]

\[ \text{jointRead } = (s.\text{citizen} \subseteq o.\text{JOINT}) \lor (o.\text{citizen} = \{\text{US}\}) \]

\[ \text{reltoRead } = (s.\text{citizen} \subseteq o.\text{RELTO}) \lor (o.\text{citizen} = \{\text{US}\}) \]

\[ \text{noformRead } = (o.\text{citizen} = \{\text{US}\}) \]

The rules fgiRead, jointRead, reltoRead, and nofornRead all look to the nationality attribute of the subject. In all four cases, US citizens are granted read access. Limitations are placed on foreign subjects, however, as only citizens of specified countries in the FGI, JOINT, or RELTO
attribute can be allowed access. If a subject has a citizenship to a country not in one of these attributes, read access must be denied. Finally, the last four read rules:

\[
\begin{align*}
\text{sarRead} &= (o.SAR \subseteq s.SAR) \\
\text{rdRead} &= (o.RD = \text{true}) \\
\text{frdRead} &= (o.FR = \text{true}) \\
\text{unciRead} &= (o.UNCI = \text{true})
\end{align*}
\]

The rules for write access start with the strict *-property where subjects are allowed to write to objects at only the same classification and compartments:

\[
\begin{align*}
\text{classWrite} &= (o.class = s.clear) \\
\text{compsWrite} &= (o.comps = s.comps)
\end{align*}
\]

With write access, FGI information cannot be written to by US citizens, while JOINT and NOFORN objects should be writable by US citizens:

\[
\begin{align*}
\text{fgiWrite} &= (s.citizen \subseteq o.FGI) \\
\text{jointWrite} &= (s.citizen \subseteq o.JOINT) \lor (o.citizen = \{\text{US}\}) \\
\text{reltoWrite} &= (s.citizen \subseteq o.RELTO) \lor (o.citizen = \{\text{US}\}) \\
\text{nofornWrite} &= (o.citizen = \{\text{US}\})
\end{align*}
\]

SAR write is similar to the MAC compartments consideration. If the subject has access to other SARs, a potential leak of information between programs exists, so only objects with matching SARs are allowed to be written to:

\[
\begin{align*}
\text{sarWrite} &= (o.SAR = s.SAR) \\
\text{rdWrite} &= (o.RD = \text{true}) \\
\text{frdWrite} &= (o.FR = \text{true}) \\
\text{unciWrite} &= (o.UNCI = \text{true})
\end{align*}
\]

The lone rule based on the environment attribute date checks that the classification of the object has not yet expired. This rule is necessary to prevent automatically declassified information from being written to after it is declassified or downgraded.

\[
\text{dateWrite} = (e.now < o.declass)
\]

5. Discussion

Government is a likely target for the use of a model such as described above. It strives to describe complex relationships between subjects and their “need-to-know” and object upon which they operate, while maintaining the basic principles of a MAC model and introducing the flexibility of an ABAC model.

This model improves over a strictly MAC environment in the flexibility afforded by adopting an ABAC approach. In a strictly MAC environment, all attributes for every entity involved would be mandatory – objects would be required to bear not only a classification and compartments, but also a value for every other possible categorization. Any changes to the possible labels on information, such as the addition of a new label, requires an update to every object and subject on the system as the new label, like all others, is mandatory.

Advantages over a strictly ABAC environment include strict rules on clearance/classification and compartments for each subject and object. In a generalized ABAC environment without this restraint, the potential exists for the introduction of objects without a classification. Without this classification we cannot ensure the simple security and *-properties apply.

6. Conclusion

Recognizing traditional access control methods as being composed of a number of attributes eases the transition to ABAC approaches to a wide range of environments. Incorporating the appropriate attributes and rules allows maintaining the strengths of traditional models, while leveraging the flexibility of ABAC.

Future work in this area includes refining the combined MAC and ABAC model and examining the impact of ABAC on MAC phenomena such as polyinstantiation.

7. References


