

Cryptographic Enforcement of Role-Based Access Control

Jason Crampton

Information Security Group, Royal Holloway, University of London
jason.crampton@rhul.ac.uk

Abstract. Many cryptographic schemes have been designed to enforce information flow policies. However, enterprise security requirements are often better encoded, or can only be encoded, using role-based access control policies rather than information flow policies. In this paper, we provide an alternative formulation of role-based access control that enables us to apply existing cryptographic schemes to core and hierarchical role-based access control policies. We then show that special cases of our cryptographic enforcement schemes for role-based access control are equivalent to cryptographic enforcement schemes for temporal access control and to ciphertext-policy and key-policy attribute-based encryption schemes. Finally, we describe how these special cases can be extended to support richer forms of temporal access control and attribute-based encryption.

1 Introduction

In some situations, we may wish to use cryptographic techniques to enforce some form of access control. Such techniques are useful when data objects have the following characteristics: read often, by many users; written once, or rarely, by the owner of the data; and transmitted over unprotected networks. Fu *et al.* [15] cite content distribution networks, such as Akami and BitTorrent, as examples where some kind of cryptographic access control is particularly suitable. In such circumstances, protected data (objects) are encrypted and authorized users are given the appropriate cryptographic keys. When cryptographic enforcement is used, the problem we must address is the efficient and accurate distribution of encryption keys to authorized users.

In recent years, there has been a considerable amount of interest in *key encrypting* or *key assignment* schemes. In such schemes, a user is given a secret value – typically a single key – which enables the user to derive some collection of encryption keys which decrypt the objects for which she is authorized. In most schemes, key derivation is performed using the secret value and some information made publicly available by the scheme administrator.

Key assignment schemes are typically used to enforce information flow policies. Unfortunately, information flow policies represent a relatively small proportion of the access control policies that we may wish to enforce. In contrast, role-based access control policies can be used to encode a wide variety of access

control requirements, but no cryptographic enforcement mechanism exists for such policies.

In this paper, we consider the cryptographic enforcement of role-based access control policies. Our main contributions are

- to provide a new characterization of role-based access control policies;
- to demonstrate how this characterization leads to an interpretation of role-based access control policies as an “authorization poset” (which is *not* the same as a role hierarchy);
- to illustrate how cryptographic enforcement of role-based access control can be used both to enforce and extend the scope of temporal access control policies [4, 5, 21] and attribute-based encryption [9, 16].

Collectively, we may regard our contributions as providing some kind of unification of several different “flavors” of policies for controlling read access to protected resources, in the sense that information flow policies, temporal access control policies and attribute-based encryption policies are all special cases of our characterization of role-based access control.

In the next section, we introduce some relevant background material. In Sections 3 and 5, we describe, respectively, how core and hierarchical role-based access control can be enforced using cryptographic techniques. In Sections 4 and 6, we consider the application of our techniques to temporal access control and attribute-based encryption, respectively. There is no separate section on related work; instead, we discuss relevant prior research, where appropriate, in Sections 2, 4 and 6.

2 Background

In this section, we provide a brief introduction to information flow policies and their enforcement using cryptographic techniques. We will apply these methods to role-based access control in Sections 3 and 5. We conclude the section with a summary of role-based access control.

2.1 Information Flow Policies

Reading an object can be interpreted as causing information to flow from the object (to the reader). An *information flow policy* specifies which flows of information are authorized [14]. To specify such a policy, we define:

- a partially ordered set (*poset*) of security labels (L, \leq) ;¹
- a set of users U and a set of objects O ;
- a security function $\lambda : U \cup O \rightarrow L$, where $\lambda(x)$ denotes the security label of entity x .

Then a user u is authorized to read o if and only if $\lambda(u) \geq \lambda(o)$. Informally, information can only flow “upwards” with respect to L ; in particular, information cannot flow from an object to a less privileged user.

¹ In other words, \leq is a reflexive, anti-symmetric, transitive, binary relation defined on L .

2.2 Cryptographic Enforcement of Information Flow Policies

The study of cryptographic enforcement of information flow policies began with the work of Akl and Taylor [1]. The basic idea is very simple. Given a set of security labels L and a function λ :

1. associate a cryptographic key $\kappa(x)$ with each $x \in L$,
2. encrypt every object o such that $\lambda(o) = x$ with $\kappa(x)$, and
3. for every user u such that $\lambda(u) = x$, ensure that u possesses, or can derive, key $\kappa(y)$ for all $y \leq x$.

Then user u can decrypt any object o with security label $y \leq x$, but cannot decrypt any other object.

There are several generic techniques that can be used to realize such cryptographic enforcement schemes for information flow [11], which are often called *key assignment schemes*. In this paper, we focus on key assignment schemes that use iterative key derivation (which is defined by the traversal of some path in an appropriate directed, acyclic graph). In such schemes, each user is given a single key and the scheme administrator publishes additional information that enables the derivation of other keys. Specifically, given a directed graph $G = (V, E)$ and a function $\lambda : U \cup O \rightarrow V$, we define a key $\kappa(x)$ for each $x \in V$, and for each edge $(x, y) \in E$ we include $\text{Enc}_{\kappa(x)}(\kappa(y))$ in the public information (where $\text{Enc}_k(m)$ denotes the symmetric encryption of message m with key k). Then a user u such that $\lambda(u) = x$ can derive the key for any $y \in V$ if there exists a directed path from x to y in G . Clearly, such a scheme can be used to enforce an information flow policy, where G is the Hasse diagram [12] of (L, \leq) .

There are two notions of security for key assignment schemes: *key recovery* and *key indistinguishability* [3, 5]. Informally, security against key recovery means that an adversary has a negligible probability of deriving a key $\kappa(x)$ if the adversary does not know $\kappa(y)$ for some $y > x$. And a scheme with key indistinguishability means that a computationally-bounded adversary A has no significantly better strategy than to guess in deciding whether a given string equals $\kappa(x)$ (where x is known to A) or is some random string. Key indistinguishability is analogous to ciphertext indistinguishability (under chosen plaintext attack) [19].

The method of encrypting the keys of child nodes using the keys of parent nodes (described above) results in a key assignment scheme secure against key recovery (assuming the encryption method is chosen appropriately). However, the use of $\kappa(x)$ to encrypt both objects and subordinate keys means that it is trivial to distinguish $\kappa(x)$ from a random string. To obtain a scheme with the key indistinguishability property we need to introduce an additional “key-encrypting key” $K(x)$ for each node x . Then we publish $\text{Enc}_{K(x)}(\kappa(x))$ for all $x \in L$ and $\text{Enc}_{K(x)}(K(y))$ for all edges $(x, y) \in E$: for $y < x$, $K(x)$ is used to derive $K(y)$ and $K(y)$ is used to derive the (“object-encrypting”) key $\kappa(y)$; crucially, $\kappa(x)$ cannot be used to derive $\kappa(y)$.

Space does not permit a detailed account of these concepts; the interested reader is referred to the literature [3, 6, 13] for further details. For our purposes, it is sufficient to note that key assignment schemes that are secure against key

recovery and have the property of key indistinguishability can be constructed for any directed, acyclic graph $G = (V, E)$.

2.3 Role-Based Access Control

The basic principles of role-based access control (RBAC) are very simple [22]. We assume that there is a set of roles that are authorized to perform certain actions and that users are authorized to “play” certain roles. The indirection between users and authorized actions provided by the set of roles means that the management of access control policies is greatly simplified.

More formally, we assume the existence of set of users U , a set of roles R and a set of permissions P (where a permission is an object-action pair). Then in *core RBAC* [2] (equivalently, the RBAC_0 model [22]) an access control policy is specified by a user-role assignment relation $UA \subseteq U \times R$ and a permission-role assignment relation $PA \subseteq P \times R$. A user u is *authorized* for permission p if there exists a role $r \in R$ such that $(u, r) \in UA$ and $(p, r) \in PA$.

An additional level of indirection may be introduced (with further reduction in management overheads) by defining a role hierarchy as a partial order relation on R (*hierarchical RBAC* [2] or RBAC_1 [22]). In this case, a user u is authorized for permission p if there exist roles r and r' such that $(u, r) \in UA$, $r \geq r'$ and $(p, r') \in PA$.

3 Cryptographic Role-Based Access Control

We first consider an alternative, but equivalent, formalism for core RBAC. (We extend our formalism to hierarchical RBAC in Sect. 5.) First let us assume that we only wish to control *read* access to objects (as is usual when considering cryptographic access control). With this assumption, there is a one-to-one correspondence between permission and objects and we may replace the set of permissions P with the set of objects O .

Given a set of roles X , then, we may represent a core RBAC policy as a function $\phi : U \cup O \rightarrow 2^X$. We interpret $\phi(u)$, where $u \in U$, as the set of roles for which u is authorized, and $\phi(o)$, where $o \in O$, as the set of roles that are authorized for o . Then (by definition) $u \in U$ is authorized for $o \in O$ if and only if $\phi(u) \cap \phi(o) \neq \emptyset$.

Note that this formulation of RBAC is rather similar to that for information flow policies, with 2^X comprising the set of security labels. However, the authorization semantics for information flow and RBAC are rather different, which means that we cannot apply the cryptographic enforcement techniques (discussed in Sect. 2) directly to the poset $(2^X, \subseteq)$. Nor can we use the binary relation \sim , defined on 2^X , where $A \sim B$ if and only if $A \cap B \neq \emptyset$, since it is neither anti-symmetric nor transitive.

We now consider what it means for an object o to be assigned to two roles r_1 and r_2 . Then any user assigned to any set of roles containing r_1 or r_2 is authorized for o . Hence, from the perspective of authorization, we may interpret

$\phi(o) = \{r_1, r_2\}$ as a “disjunction” of the roles $r_1 \vee r_2$. A similar argument suggests that $\phi(u) = \{r_1, r_2\}$ should be interpreted as a “conjunction” $r_1 \wedge r_2$.

With this insight, we use the set X as a set of “atoms” to construct an “authorization poset” $\text{Auth}(X, \phi)$, where $x \wedge y$ represents the authorization label for any user u such that $\phi(u) = \{x, y\}$ and $x \vee y$ represents the authorization label of any object o such that $\phi(o) = \{x, y\}$.² Henceforth, we write $\bigvee \{a_1, \dots, a_k\}$ to denote $a_1 \vee \dots \vee a_k$ and $\bigwedge \{a_1, \dots, a_k\}$ to denote $a_1 \wedge \dots \wedge a_k$. We now define the authorization poset induced by a core RBAC policy.

Definition 1 Let $\phi : U \times O \rightarrow 2^X$ define a core RBAC policy, where X is a set of roles. Then we define $(\text{Auth}(X, \phi), \sqsubseteq)$, the authorization poset induced by ϕ and X , in the following way:

- for all $x \in X$, $x \in \text{Auth}(X, \phi)$;
- if $A = \phi(u)$ for some $A \in 2^X$ and some $u \in U$, then $\bigwedge A \in \text{Auth}(X, \phi)$;
- if $B = \phi(o)$ for some $B \in 2^X$ and some $o \in O$, then $\bigvee B \in \text{Auth}(X, \phi)$;
- $\bigvee A \sqsubseteq \bigvee B$ if and only if $A \supseteq B$;
- $\bigwedge A \sqsubseteq \bigwedge B$ if and only if $A \subseteq B$;
- $\bigvee A \sqsubseteq \bigwedge B$ if and only if $A \cap B \neq \emptyset$;
- $\bigwedge A \not\sqsubseteq \bigvee B$.

Henceforth, we will omit ϕ from $\text{Auth}(X, \phi)$ as it will always be obvious from context. Note that $x \in X$ can be represented either as $\bigvee \{x\}$ or $\bigwedge \{x\}$, and these representations are vacuously equivalent from the perspective of authorization. Hence, we omit these duplication representations and, for all $x \in X$, we simply include x in $\text{Auth}(X)$. Figure 1 illustrates $\text{Auth}(X)$ for $X = \{a, b, c\}$ (under the assumption that for all $A \subseteq X$, there exists $u \in U$ such that $\phi(u) = A$ and there exists $o \in O$ such that $\phi(o) = A$).

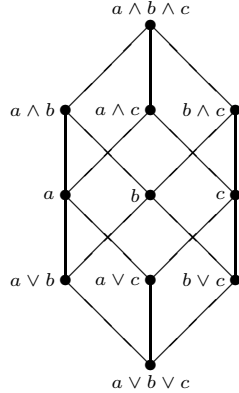


Fig. 1. $(\text{Auth}(X), \sqsubseteq)$ for $X = \{a, b, c\}$

² It is important to stress that $x \wedge y$ is simply notation that is intended to convey some intuition about how the assignment of roles x and y to a user should be interpreted; it does not represent the logical conjunction of x and y . The same is true for $x \vee y$.

Proposition 2 Let $\phi : U \cup O \rightarrow X$ be a core role-based access control policy and let $|X| = n$. Then $(\text{Auth}(X), \sqsubseteq)$ is a poset and $|\text{Auth}(X)| \leq 2^{n+1} - n - 2$.

Proof. We have to establish that \sqsubseteq is reflexive, anti-symmetric and transitive. Recall that \subseteq and \supseteq (subset and superset inclusion, respectively) define partial orders on 2^X for any set X .

- Then \sqsubseteq is reflexive, since any element in $\text{Auth}(X)$ has the form $\bigwedge A$ or $\bigvee A$ for some $A \subseteq X$, and reflexivity of \sqsubseteq therefore follows from the reflexivity of \subseteq and \supseteq , respectively.
- Suppose that $C \sqsubseteq D$ and $D \sqsubseteq C$. Then, without loss of generality, it cannot be the case that $C = \bigwedge A$ and $D = \bigvee B$ for some $A, B \subseteq X$, since, by definition, $\bigwedge A \not\sqsubseteq \bigvee B$ for all $A, B \subseteq X$. Hence, either $C = \bigwedge A$ and $D = \bigwedge B$ or $C = \bigvee A$ and $D = \bigvee B$ for some $A, B \subseteq X$. The anti-symmetry of \sqsubseteq then follows immediately from the anti-symmetry of \subseteq and \supseteq .
- Suppose that $D \sqsubseteq E$ and $E \sqsubseteq F$. First, consider the case where $D = \bigwedge A$, $E = \bigwedge B$ and $F = \bigwedge C$ for $A, B, C \subseteq X$. Then $D \sqsubseteq F$ by the transitivity of \subseteq . The same is true if $D = \bigvee A$, $E = \bigvee B$ and $F = \bigvee C$. Now consider the case where $D = \bigvee A$, $E = \bigwedge B$ and $F = \bigwedge C$. Then, by definition, there exists $x \in A \cap B$, and $x \in C$, since $B \subseteq C$. Hence, $D \sqsubseteq F$. Clearly, a similar line of argument can be used if $D = \bigvee A$, $E = \bigvee B$ and $F = \bigwedge C$. (Note that we can discount cases like $D = \bigwedge A$, $E = \bigvee B$ and $F = \bigwedge C$, as we did in the proof of the anti-symmetric property, because, by definition, $E \not\sqsubseteq F$.)

By definition $\text{Auth}(X)$ may contain up to two copies of each subset of X of cardinality greater than 1 and one copy of each singleton subset of X . There are $2^n - (n + 1)$ subsets of X of cardinality greater than 1 and there are n singleton subsets of X . Hence, $|\text{Auth}(X)| \leq 2(2^n - (n + 1)) + n$. The result follows. \square

Note that $(\text{Auth}(X), \sqsubseteq)$ is not a lattice, even though its “building blocks” – $(2^X, \subseteq)$ and $(2^X, \supseteq)$ – are lattices, because some pairs of elements (a and $b \vee c$ in Fig. 1, for example) do not have a unique least upper bound.

Proposition 3 Let X , UA and PA define a core RBAC policy, and let $(\text{Auth}(X), \sqsubseteq)$ be the authorization poset induced by X and ϕ . Then user $u \in U$ is authorized for object $o \in O$ with respect to the usual core RBAC semantics if and only if $\bigvee \phi(o) \sqsubseteq \bigwedge \phi(u)$.

Proof. u is authorized for object o if and only if there exists $r \in X$ such that $(u, r) \in UA$ and $(p, r) \in PA$; that is, $r \in \phi(u) \cap \phi(o)$. And $r \in \phi(u) \cap \phi(o)$ if and only if $\bigvee \phi(o) \sqsubseteq \bigwedge \phi(u)$. \square

In other words, if we have a collection of objects to which read access should be restricted according to some RBAC policy ϕ with role set X , then we can define a collection of cryptographic keys, one for each element of $\text{Auth}(X)$. Given that $\text{Auth}(X)$ is a poset, we can now use existing key assignment schemes to generate a set of public information for $(\text{Auth}(X), \leq)$: that is, if $x \sqsupseteq y$ in $\text{Auth}(X)$, then the key associated with $x \in \text{Auth}(X)$, denoted $\kappa(x)$, can be used to derive $\kappa(y)$. We refer to this as a *cryptographic role-based access control* (CRBAC) scheme.

Note that, given $\kappa(\bigwedge A)$, it is only necessary to be able to derive $\kappa(a)$, $a \in A$, because no object is encrypted with any key of the form $\kappa(\bigwedge B)$, where $B \subseteq A$. From this observation, we deduce the following result.

Proposition 4 *Let $\phi : U \cup O \rightarrow X$ define a core RBAC policy, and let $|X| = n$. Then there exists a set of edges $E \subseteq \text{Auth}(X) \times \text{Auth}(X)$ such that $|E| \leq n(2^n - 2)$ and the diameter³ of the graph $(\text{Auth}(X), E)$ (and hence the number of key derivation steps) is no greater than 2.*

Proof. We construct E in the following way:

- for a node $\bigwedge A$, where $A = \phi(u)$ for some $u \in U$, we add an edge between node $\bigwedge A$ and node a for all $a \in A$, and
- for node $\bigvee B$, where $B = \phi(o)$ for some $o \in O$, we add an edge between b and $\bigvee B$ for all $b \in B$.

Now each subset $A \in 2^X$ such that $|A| > 1$ contributes at most $2|A|$ edges ($|A|$ edges for each of the elements $\bigvee A$ and $\bigwedge A$). Hence,

$$|E| \leq 2 \sum_{i=2}^n i \binom{n}{i} = 2 \left(\sum_{i=1}^n i \binom{n}{i} - n \right) = 2n \left(\sum_{i=1}^n \binom{n-1}{i-1} - 1 \right) = 2n(2^{n-1} - 1)$$

Finally, $\bigwedge A$ should be able to derive the key for $\bigvee B$ only if there exists $x \in A \cap B$. Now, by construction, there exists an edge $(\bigwedge A, x)$ and an edge $(x, \bigvee B)$ and key derivation takes precisely two hops. \square

In Sec. 5, we consider how we can incorporate role hierarchies into our work. Before that, we consider a class of access control policies that can be interpreted as instances of core RBAC policies in which we are interested in particular subsets of 2^X and can ignore certain “conjunctions” and “disjunctions”.

4 Application: Temporal Access Control

In recent years, we have seen the development of access control models in which time plays an important role in deciding whether access requests are authorized or not [7]. One particular application of such “temporal access control” systems is the protection of data that is made available periodically as (part of) a subscription-based service [8]. In this section, we consider an application of CRBAC in which we extend the scope of temporal access control policies and their enforcement using cryptographic techniques. We do this by restricting attention to particular families of subsets of some set of roles.

In particular, each time interval (a consecutive sequence of time points) represents a security label that may be assigned to a user, and each time point represents a security label that may be assigned to an object. Then objects are encrypted using the key associated with a particular time point t and a user associated with time interval $[x, y]$ is authorized to access the object if $t \in [x, y]$.

³ the length of the longest path contained in the graph

Figure 2(a) illustrates the poset when the number of time points is 4. Atallah *et al.* [4] and Ateniese *et al.* [5] have constructed key assignment schemes for temporal access control using this model.

Alternatively, we may wish to define policies in which objects are associated with a time interval and users are associated with a particular time point. We may, for example, wish to release an encrypted object at time t in such a way that it can be decrypted by keys $\kappa(t), \kappa(t+1), \dots, \kappa(t+d)$ (where $\kappa(t)$ denotes the key associated with time point t). Paterson and Quaglia recently introduced the notion of *time-specific encryption* [21], which uses this model of temporal access control, and also discussed potential applications of time-specific encryption. In this case, it is users that are assigned to single time points and objects that are assigned to intervals.

It is easy to see from the above observations that CRBAC can be used to enforce these two different interpretations of temporal access control. In the first case, we are only interested in labels of the form $\bigwedge[t_i, t_j]$ (since objects are only associated with single time points); in the second case, we only consider labels of the form $\bigvee[t_i, t_j]$ (since users are only associated with single time points). The posets representing these two situations for $X = \{t_1, t_2, t_3, t_4\}$ are shown in Fig. 2.

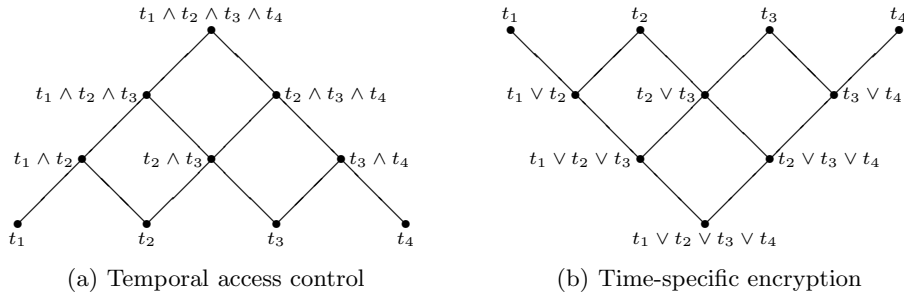


Fig. 2. Authorization posets for temporal access control and time-specific encryption

More generally, we may wish to assign both users and objects to intervals. Then an object associated with interval $[x, y]$ can be accessed by any user associated with interval $[x', y']$, where $x' \leq y$ (in other words, whenever the intervals overlap). In this context, we define two types of intervals.

- We write $[x \wedge y]$ to denote an interval that can only be assigned to users. A user with key $\kappa([x \wedge y])$ indicates that the user is authorized for all time points $t \in [x, y]$;
- An interval of the form $[x \vee y]$ can only be assigned to objects and indicates the object is associated with each of the time points $t \in [x, y]$. In particular, $\kappa(t)$ can decrypt an object encrypted with $\kappa([x \vee y])$ for all $t \in [x, y]$.

In this context, the set of security labels forms a diamond-shaped grid, where $[x \wedge y] \geq [z, z]$ and $[z, z] \geq [x \vee y]$ for all $z \in [x, y]$.

5 Role Hierarchies

In hierarchical RBAC, the set of roles is partially ordered. In the hierarchical setting, we must therefore provide an appropriate interpretation of $r_1 \wedge r_2$ and $r_1 \vee r_2$. There are two situations to consider:

- $r_1 \leq r_2$ (and, equivalently, $r_1 \geq r_2$);
- r_1 and r_2 are incomparable (that is, $r_1 \not\leq r_2$ and $r_1 \not\geq r_2$), which we abbreviate henceforth to $r_1 \parallel r_2$.

If $r_1 \leq r_2$, then any user assigned to r_2 is also (implicitly) assigned to r_1 , and any permission assigned to r_1 is also assigned to r_2 . Therefore, $r_1 \wedge r_2$ is equivalent, in terms of authorization, to r_2 , and $r_1 \vee r_2$ is equivalent to r_1 . In contrast, $r_1 \wedge r_2$ and $r_1 \vee r_2$ cannot be simplified if $r_1 \parallel r_2$. Generalizing, we have:

- for every non-empty chain $C \subseteq X$, $\bigvee C$ is the minimum element in C and $\bigwedge C$ is the maximum element in C ;
- for every non-empty antichain $A \subseteq X$, $\bigvee A$ and $\bigwedge A$ cannot be simplified.⁴

Note that the above discussion means we may assume that the set of roles for which a user is authorized is an antichain and the set of roles that are authorized for a permission also forms an antichain. We write \mathcal{A}^X to denote the set of antichains in a poset (X, \leq) . The following result (stated without proof) establishes the existence of two partial orders on \mathcal{A}^X ; these orderings are analogous to subset and superset inclusion for a powerset.

Lemma 5 (Crampton [10]) *Let (X, \leq) be a poset, and let \mathcal{A}^X be the set of antichains in X . For $A, B \in \mathcal{A}^X$, we define:*

- $A \preceq_1 B$ if and only if for all $b \in B$ there exists $a \in A$ such that $a \leq b$, and
- $A \preceq_2 B$ if and only if for all $a \in A$ there exists $b \in B$ such that $a \leq b$.

Then \preceq_1 and \preceq_2 define partial orders on \mathcal{A}^X .

Definition 6 *Given a poset of roles (X, \leq) and a hierarchical RBAC policy $\phi : U \cup O \rightarrow \mathcal{A}^X$, we define $(\text{Auth}(X), \sqsubseteq)$, the authorization poset induced by (X, \leq) and ϕ , in the following way:*

- for all $x \in X$, $x \in \text{Auth}(X)$;
- if $A = \phi(u)$ for some $A \in \mathcal{A}^X$, then $\bigwedge A \in \text{Auth}(X)$;
- if $B = \phi(o)$ for some $B \in \mathcal{A}^X$, then $\bigvee B \in \text{Auth}(X)$;
- $\bigvee A \sqsubseteq \bigvee B$ if and only if $A \preceq_1 B$;
- $\bigwedge A \sqsubseteq \bigwedge B$ if and only if $A \preceq_2 B$;
- $\bigvee A \sqsubseteq \bigwedge B$ if and only if there exists $x \in X$ such that $A \preceq_1 \{x\}$ and $\{x\} \preceq_2 B$;
- $\bigwedge A \not\sqsubseteq \bigvee B$.

Note that an information flow policy for confidentiality is a trivial special case in which (X, \leq) represents the security lattice and for all $u \in U$ and $o \in O$, $\phi(u)$ and $\phi(o)$ are singleton sets. The construction of $\text{Auth}(X)$ for a simple poset of roles X is shown in Fig. 3. (The figure assumes that every antichain appears exactly twice.)

⁴ Note that in core RBAC, we may take the partial order on R to be the empty set, and every subset of R is an antichain.

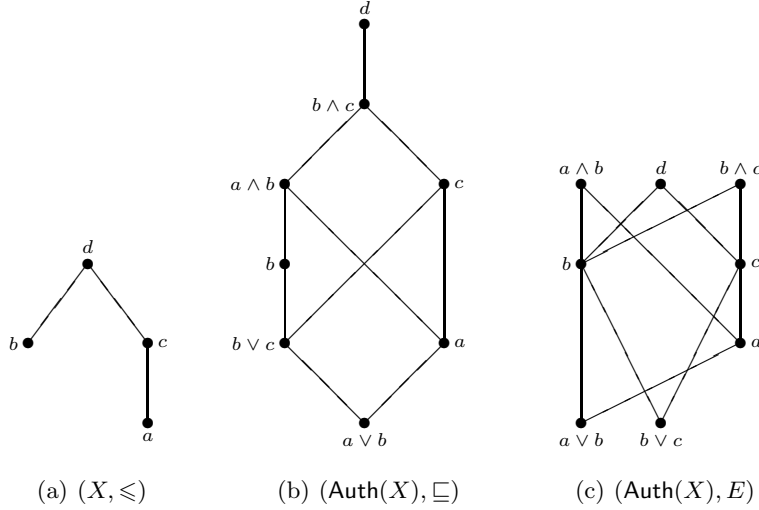


Fig. 3. $\text{Auth}(X)$ for a simple poset X

We have two results analogous to those for core RBAC. The proof of each result is very similar to the respective proofs in Sect. 3 and rely on Lemma 5. The orderings \preceq_1 (which is used to define the ordering on elements of the form $\bigvee A$) and \preceq_2 (used for elements of the form $\bigwedge A$) replace \supseteq and \subseteq , respectively. The proofs of the following results are omitted due to space constraints.

Proposition 7 *Let (X, \leq) be a poset. Then $(\text{Auth}(X), \subseteq)$ is a poset.*

Proposition 8 *Let (X, \leq) , UA and PA define a hierarchical RBAC policy, and let $\text{Auth}(X)$ be the authorization lattice induced by X and ϕ . Then user $u \in U$ is authorized for object $o \in O$ with respect to the usual hierarchical RBAC semantics if and only if $\bigvee \phi(o) \subseteq \bigwedge \phi(u)$.*

We cannot prove a general result analogous to Proposition 4 because the cardinality of $\text{Auth}(X)$ depends on the antichains in X (which depend on the partial order). However, we can provide an upper bound on the number of nodes as a function of the number of antichains in X , which we denote by a . Then there are $a - n$ antichains of cardinality greater than 1. Therefore, by construction, $|\text{Auth}(X)| \leq 2(a - n) + n = 2a - n$. (Note that when X is unordered, every non-empty subset is an antichain, and we recover the result in Proposition 4.)

We can also construct a set of edges such that the diameter of $(\text{Auth}(X), E)$ is 2. We construct a set of edges E in the following way:

- if $A = \phi(u)$ for some $u \in U$, then $(\bigwedge A, a) \in E$ for all $a \in A$,
- if $B = \phi(o)$ for some $o \in O$, then $(b, \bigvee B) \in E$ for all $b \in B$, and
- if (x, y) is an edge in the (graph of the) transitive reduction of (X, \leq) , then $(x, y) \in E$.

Clearly the diameter of the graph $(\text{Auth}(X), E)$ is 2, as in the proof of Proposition 4. Like $|\text{Auth}(X)|$, the cardinality of E depends on a . In general,

$$|E| \leq m + 2 \sum_{A \in \mathcal{A}^X} |A|,$$

where m is the cardinality of the covering relation of (X, \leq) . An example of the graph $(\text{Auth}(X), E)$ is shown in Fig. 3(c).

6 Application: Attribute-Based Encryption

In this section, we examine the connections between CRBAC and *attribute-based encryption* (ABE). In *ciphertext policy ABE* (CP ABE) a message is associated with multiple attribute sets, whereas a user (who is given a private decryption key) is associated with a single attribute set [9].

More specifically, in CP ABE we assume the existence of a set of attributes Att . Each encryption key is associated with a *monotone access structure* \mathcal{S} defined over Att . That is, \mathcal{S} is a collection of subsets of Att such that if $A \in \mathcal{S}$ and $A \subseteq B$, where $A, B \subseteq \text{Att}$, then $B \in \mathcal{S}$.⁵ In contrast, each decryption key is associated with some subset of Att . The keys are constructed in such a way that a message encrypted with k , where k is associated with monotone access structure \mathcal{S} , can only be decrypted by k' , where k' is associated with $A \subseteq \text{Att}$, if $A \in \mathcal{S}$.

We assume that users are synonymous with decryption keys and objects are synonymous with encryption keys. Then an instance of CP ABE is completely defined by a function $\psi : U \cup O \rightarrow 2^{2^{\text{Att}}}$, where

- $\psi(u)$, $u \in U$, is equal to $\{A\}$ for some $A \subseteq \text{Att}$;
- $\psi(o)$, $o \in O$, is equal to some monotone access structure \mathcal{S} defined over Att .

A user u is, by definition, authorized for object o if $\psi(u) \in \psi(o)$. Clearly, we may encode an instance of CP ABE as an instance of a *core* RBAC policy, in which $X = 2^{\text{Att}}$, $\phi(u) = \psi(u)$ and $\phi(o) = \bigvee \psi(o)$. By definition, u is authorized for o if $\phi(u) \cap \phi(o) \neq \emptyset$ and, since $\phi(u)$ is a singleton set in 2^{Att} , this condition holds if and only if $\phi(u) \in \phi(o)$.

However, we can encode ψ more economically as a hierarchical role-based access control policy. In particular, we define the set of roles to be $(2^{\text{Att}}, \subseteq)$ and we define

- $\phi(u) = \psi(u)$;
- $\phi(o) = \bigvee [\psi(o)]$, where $[\mathcal{S}]$ denotes the set of minimal elements in the monotone access structure \mathcal{S} .⁶

Figure 4 illustrates the authorization lattice induced by the hierarchical RBAC interpretation of CP ABE when $|\text{Att}| = 3$. To prove that this hierarchical RBAC policy does indeed encode ψ , we establish the following result.

⁵ Equivalently, \mathcal{S} is an *order filter* [12] in the poset $(2^{\text{Att}}, \subseteq)$.

⁶ A set A_i is minimal in the collection of sets $\{A_1, \dots, A_k\}$ if there does not exist A_j such that $A_j \subset A_i$. It is easy to show that a monotone access structure is uniquely defined by its minimal elements (see [10], for example).

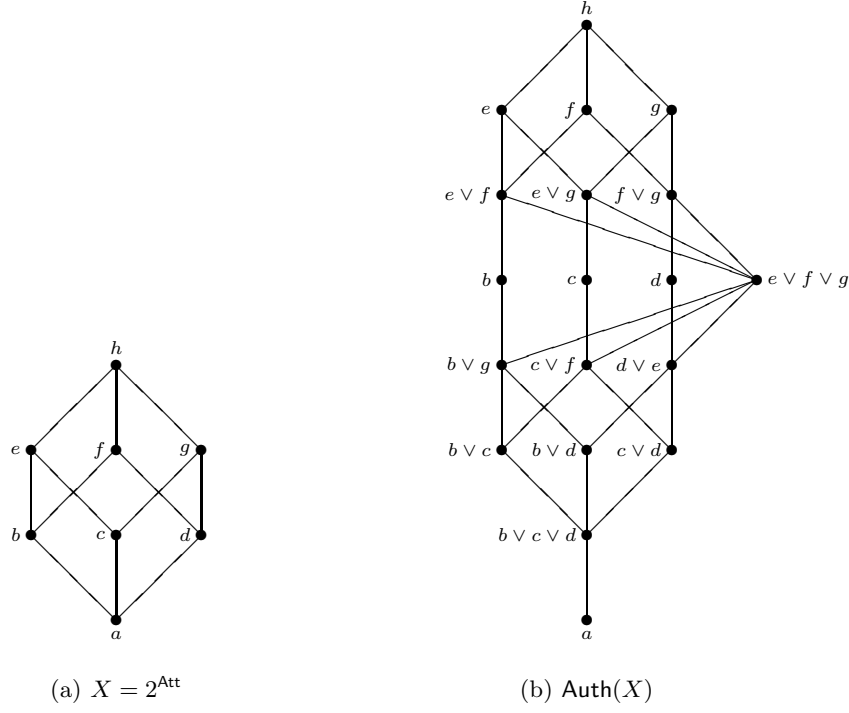


Fig. 4. The authorization poset for CP ABE when $|\text{Att}| = 3$

Proposition 9 *Let ψ define an instance of ciphertext policy attribute-based encryption. Then $\psi(u) \in \psi(o)$ if and only if $\bigvee [\psi(o)] \sqsubseteq \psi(u)$.*

Proof. “ \Rightarrow ”: Since $\psi(o)$ is a monotone access structure, $\psi(u) \in \psi(o)$ implies that there exists a minimal element $M \in \psi(o)$ such that $M \subseteq \psi(u)$. And, by definition, $M \in [\psi(o)]$. Hence, $\bigvee [\psi(o)] \preceq_1 \{M\} \preceq_2 \psi(u)$. That is, $\bigvee [\psi(o)] \sqsubseteq \psi(u)$.

“ \Leftarrow ”: If $\bigvee [\psi(o)] \sqsubseteq \psi(u)$, then, by definition, there exists $M \in 2^X$ such that $\bigvee [\psi(o)] \preceq_1 \{M\}$ and $\{M\} \preceq_2 \psi(u)$. Since $\psi(u)$ is a singleton set, the definition of \preceq_2 implies that $M \subseteq \psi(u)$. Moreover, $\bigvee [\psi(o)] \sqsubseteq \{M\}$ implies that there exists $M' \in [\psi(o)]$ such that $M' \subseteq M$. Hence, we may conclude that $M' \subseteq \psi(u)$, which implies that $\psi(u) \in \psi(o)$ since $\psi(o)$ is a monotone access structure and $M' \in \psi(o)$. \square

We conclude that CRBAC can be used to implement CP ABE. The crucial difference is that CRBAC uses only symmetric primitives, whereas CP ABE relies on pairing-based cryptographic primitives. Space constraints preclude a comparison of the characteristics of CP ABE and CRBAC (such as security

notions, key size, ciphertext size, encryption and decryption times); this will be the subject of future work.

In contrast to CP ABE, *key policy ABE* (KP ABE) associates decryption keys with monotone access structures and encryption keys with single attribute sets [16]. The decryption key k , where $\psi(k) = \mathcal{S}$ for some monotone access structure \mathcal{S} , can decrypt any message encrypted with k' , where $\psi(k') = B \subseteq \text{Att}$ for any $B \in \mathcal{S}$. As for KP ABE, we can show that CRBAC can be used to enforce KP ABE; we omit these details.

Note, however, that there is no particular reason why we should associate a single element of $X = 2^{\text{Att}}$ with either encryption (as in KP ABE) or decryption (as in CP ABE). In other words, we can do for attribute-based encryption exactly what we did for temporal access control in Sect. 4, and associate arbitrary antichains in 2^{Att} with both encryption and decryption keys. To our knowledge, no scheme in the literature has considered the simultaneous enforcement of CP and KP ABE. Again, space constraints do not allow us to explore this matter in appropriate detail and is something we will be pursuing in future work. Figure 5 illustrates the full authorization poset required to support CP and KP ABE when $|\text{Att}| = 3$. Notice the embedding of the poset depicted in Fig. 4 within the poset shown in Fig. 5.

7 Concluding Remarks

The main contribution of this paper is to introduce a way of re-writing an RBAC policy in such a way that it can be interpreted as an information flow policy.⁷ Having re-written an RBAC policy as an information flow policy, we can apply techniques from the literature on key assignment schemes to provide a cryptographic enforcement mechanism for the original RBAC policy.

Perhaps the most interesting aspect of our work is that several different strands of work on cryptographic enforcement of access control policies – such as temporal access control and attribute-based encryption – are special cases of our cryptographic enforcement mechanism for RBAC. This is unsurprising as RBAC is claimed to be very expressive [22], but it does illustrate the importance of our contribution in providing a cryptographic enforcement mechanism for RBAC, thereby providing a uniform cryptographic enforcement mechanism for several classes of authorization policies.

In future work, we will continue to explore the connections between our work and attribute-based encryption, with a particular emphasis on the simultaneous enforcement of ciphertext policy and key policy attribute-based encryption. Previous work on cryptographic file systems suggests that asymmetric cryptosystems are required to support read-write access control policies; consequently key management becomes more complex [17, 18]. Therefore, we also intend to investigate whether it is possible to support both read and write access modes for RBAC using the (symmetric encryption) techniques described in this paper,

⁷ It should be noted that our technique is quite different from attempts in the literature to use RBAC to enforce information flow policies (as in [20], for example).

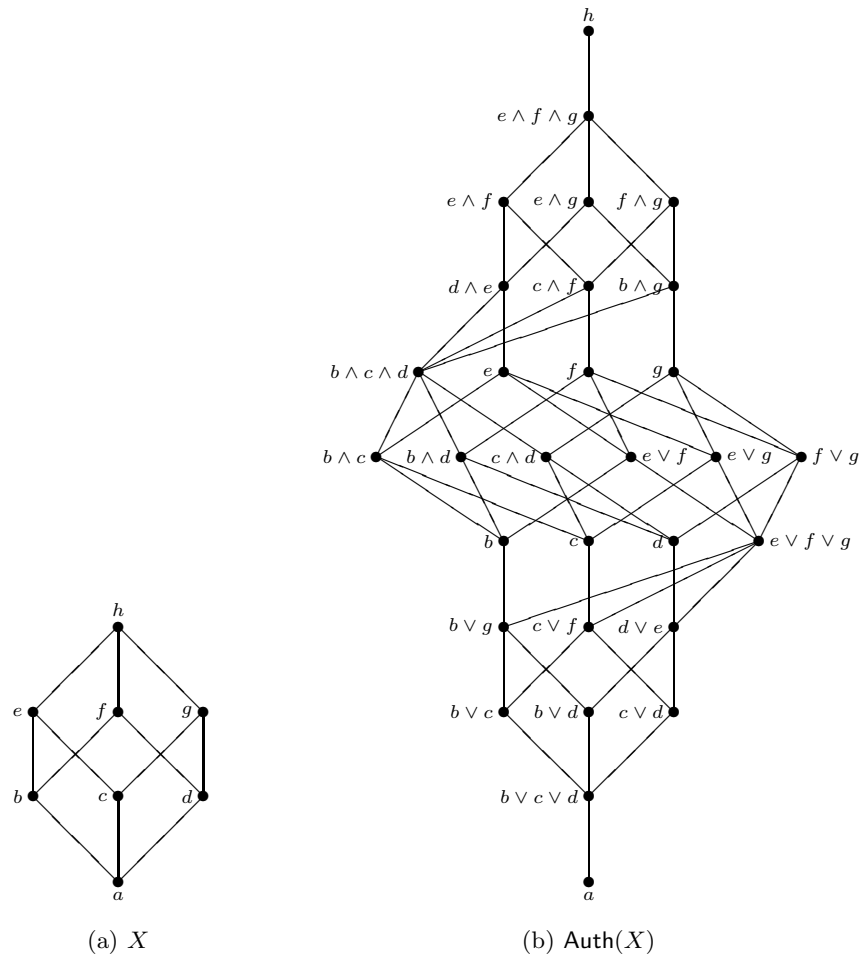


Fig. 5. $\text{Auth}(X)$ when the set of roles X is order isomorphic to a powerset

perhaps using message authentication codes, rather than digital signatures, to confirm the validity of attempts to modify an object.

References

1. Akl, S., Taylor, P.: Cryptographic solution to a problem of access control in a hierarchy. *ACM Transactions on Computer Systems* **1**(3), 239–248 (1983)
2. American National Standards Institute: *ANSI INCITS 359-2004 for Role Based Access Control* (2004)
3. Atallah, M., Blanton, M., Fazio, N., Frikken, K.: Dynamic and efficient key management for access hierarchies. *ACM Transactions on Information and System Security* **12**(3), 1–43 (2009)

4. Atallah, M., Blanton, M., Frikken, K.: Incorporating temporal capabilities in existing key management schemes. In: Proceedings of the 12th European Symposium on Research in Computer Security, pp. 515–530 (2007)
5. Ateniese, G., De Santis, A., Ferrara, A., Masucci, B.: Provably-secure time-bound hierarchical key assignment schemes. Cryptology ePrint Archive, Report 2006/225 (2006)
6. Ateniese, G., De Santis, A., Ferrara, A., Masucci, B.: Provably-secure time-bound hierarchical key assignment schemes. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 288–297 (2006)
7. Bertino, E., Bonatti, P., Ferrari, E.: TRBAC: A temporal role-based access control model. *ACM Transactions on Information and System Security* **4**(3), 191–223 (2001)
8. Bertino, E., Carminati, B., Ferrari, E.: A temporal key management scheme for secure broadcasting of XML documents. In: Proceedings of the 8th ACM Conference on Computer and Communications Security, pp. 31–40 (2002)
9. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of 2007 IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
10. Crampton, J.: Authorization and antichains. Ph.D. thesis, Birkbeck, University of London, London, England (2002)
11. Crampton, J., Martin, K., Wild, P.: On key assignment for hierarchical access control. In: Proceedings of 19th Computer Security Foundations Workshop, pp. 98–111 (2006)
12. Davey, B., Priestley, H.: Introduction to Lattices and Order, second edn. Cambridge University Press, Cambridge, United Kingdom (2002)
13. De Santis, A., Ferrara, A., Masucci, B.: Efficient provably-secure hierarchical key assignment schemes. Cryptology ePrint Archive, Report 2006/225 (2006)
14. Denning, D.: A lattice model of secure information flow. *Communications of the ACM* **19**(5), 236–243 (1976)
15. Fu, K., Kamara, S., Kohno, T.: Key regression: Enabling efficient key distribution for secure distributed storage. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2006 (2006)
16. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
17. Harrington, A., Jensen, C.: Cryptographic access control in a distributed file system. In: Proceedings of Eighth ACM Symposium on Access Control Models and Technologies, pp. 158–165 (2003)
18. Kallahalla, M., Riedel, E., Swaminathan, R., Wang, Q., Fu, K.: Plutus: Scalable secure file sharing on untrusted storage. In: Proceedings of the FAST '03 Conference on File and Storage Technologies, pp. 29–42 (2003)
19. Katz, J., Lindell, Y.: Introduction to Modern Cryptography. Chapman & Hall/CRC (2007)
20. Osborn, S., Sandhu, R., Munawar, Q.: Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Transactions on Information and System Security* **3**(2), 85–106 (2000)
21. Paterson, K., Quaglia, E.: Time-specific encryption. In: J. Garay (ed.) Proceedings of Seventh Conference on Security and Cryptography for Networks (2010). To appear
22. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-based access control models. *IEEE Computer* **29**(2), 38–47 (1996)