Modeling of Work-Based Access Control for Cooperative Healthcare Systems with XACML

Mohamed Abomhara and Henrik Nergaard

Department of Information and Communication Technology
University of Agder, Grimsatd, Norway
Email: {mohamed.abomhara, henrik.nergaard}@uia.no

***Abstract*—Access control is ideal for managing access to information and controlling legitimate user activities by mediating every user attempt to access a system resource. However, in a collaborative environment, the biggest challenges with deploying access control are deciding on the extent and limit of resource sharing as well as difficulty with editing, managing and updating access control policies. In our previous work in the area of access control, we proposed a work-based access control (WBAC) model that strikes a balance between collaboration and safeguarding sensitive patient information. The current study extends on that work by demonstrating how eXtensible Access Control Markup Language (XACML) is used to express WBAC policies. We explain the WBAC model for cooperative healthcare systems, implement the WBAC profile using XACML 2.0, specify permissions and define all authorization policies. We examine the access policies and show how the WBAC model simplifies decentralized administrative tasks (e.g., changing of team members and shifting responsibilities), thus enhancing the practicability of access control in dynamic collaboration environments.**

Keywords*–*Access control; Collaboration environments; Healthcare.

I. Introduction

Access control policies play an important role in ensuring that the information flow between authorized entities is controlled while preserving resource security in the face of inappropriate access [1]. Access control policies specify which authorized entity (e.g., user or organization) can perform what operations (e.g., read and write) on specific resources (e.g., files on electronic health records (EHRs)).

In collaborative environments such as healthcare, it is not easy for classical access control models like Role-Based Access Control (RBAC) [2] and Attribute-Based Access Control (ABCA) [3] alone to specify authorization constraints due to the complexity of a continuously growing as well as changing number of users and medical records. In addition to a lack of granularity, manageability and flexibility for the specification and maintenance of policies [4], [5]. Moreover, inconsistencies between the access control policies of various individuals or organizations are a common challenge [6]. It is important to understand whether and under what circumstances resources can be shared during collaboration and how collaboration can be achieved securely in the presence of inconsistencies between collaborating participants [7]. Changing participants is another challenge in access control [8]. Access control policies for centralized environments do not address the dynamic changes of participating groups in distributed environments.

The possibility of information leaks caused by improperly designed authorization policies consequently increases. Thus, some extra authorization constraints should be added to traditional authorization mechanisms to prevent information leaks caused by inadequately designed policies. Moreover, access control policies must be flexible and configured to control the dynamic interactions during collaboration [9], [10].

In this study, we demonstrate and implement an access control model for a collaborative healthcare environment to support diverse domains of data authorization management with various constraints. The implantation is built based on eXtensible Access Control Markup Language (XACML) [11] using a Work-Based Access Control (WBAC) model [12], [13], [14] (works by one of the current authors). The aim is to simplify decentralized administrative tasks and thus enhance the practicability of access control in dynamic collaboration environments. WBAC introduces the team role concept, and modified the user-role assignment model from previous works [15], [16]. WBAC handles access control based on collaborative work and team member assignment. The team is segregated into *strategic*, *action* and *management* groups depending on the contributions to the collaborative work.

The remaining parts of this study are structured as follows: Section II presents a background of XACML and usage scenarios of collaboration and healthcare data sharing. Section III demonstrates the modeling structures, authorization constraints, request model and policy model. Section IV presents the experiments and results. Related works, discussion, conclusions and future work recommendations are provided in Section V.

II. Background

In this Section, relevant work underlying the current study is discussed. First, the XACML framework is briefly introduced, followed by concise usage scenarios to better understand collaboration in the healthcare domain.

A. An Overview of XACML

XACML is a standardized policy language by OASIS [11]. It defines the architecture, policies and messages of an access control system. XACML is a powerful and flexible policy language for heterogeneous distributed systems and is a general-purpose access control policy language [17], [18]. According to the reference XACML architecture shown in Fig. 1, the XACML model contains the following main entities [19]:

- *The Policy Enforcement Point (PEP)* is an entity that intercepts a user's request to access a resource. The

ISBN: 978-1-61208-511-1

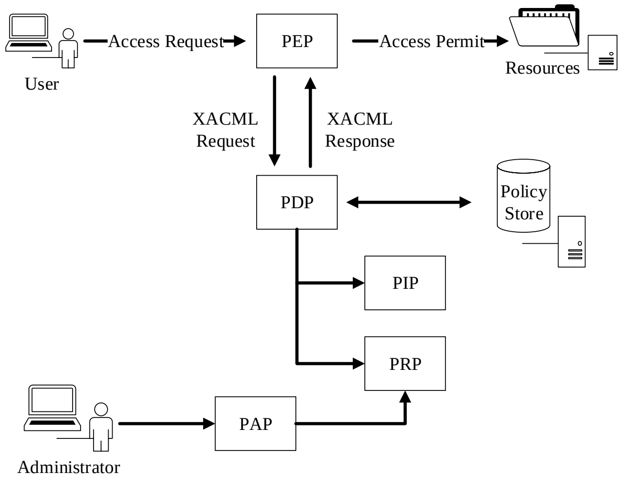


Figure 1. XACML framework

PEP forwards the request to the PDP to obtain the access decision (permit or deny). PEP then acts on the received decision.

- *The Policy Decision Point (PDP)* is used to evaluate access requests against authorization policies and makes decisions according to the information contained in the request before issuing access decisions.
- *The Policy Information Point (PIP)* acts as the source of attribute values, or the data required for policy evaluation.
- *The Policy Retrieval Point (PRP)* is an entity that stores the XACML access authorization policies.
- *The Policy Administration Point (PAP)* manages the access authorization policies.

The XACML core policy structure (Fig. 2) consists of three components: the rule, policy and policy set [19]. The rule is a fundamental component of an XACML policy. The rule, policy and policy set have a target that PDP uses to quickly find the sub-policy parts applicable to making a decision regarding an access request. The target contains a set of attribute value pairs for matching the subject, resource, action and environment, to check if the given rule, policy and policy set are applicable to a specific request. Several rules are grouped and encapsulated into policies and policies are grouped into policy sets. A rule consists of a condition and an effect that can be either a permission or denial associated with the successful evaluation of the rule. A condition represents an expression that refines the applicability of the rule beyond the predicates implied by its target. The correct evaluation of a condition returns the effect of the rule, while incorrect evaluation results in an error (*Indeterminate*) or the discovery that the condition does not apply to the request (*Not Applicable*).

PDP can use different rules, policies and policy sets to make a decision for a specific request. Therefore, conflict might occur between multiple policies when policies offer different authorization decisions. Thus, XACML provides a set of combining algorithms for combining rules and policies to solve a decision conflict between multiple policies [19]. The most commonly utilized combining algorithms are as follows:

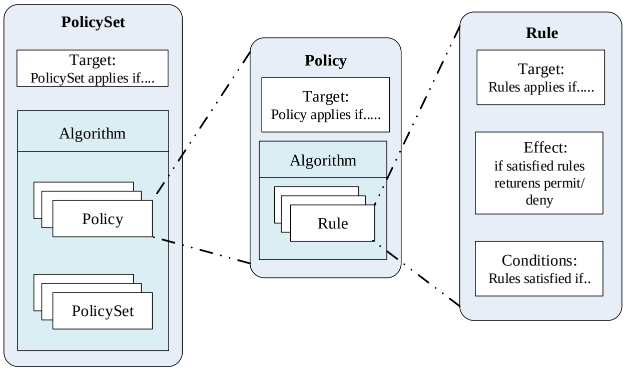


Figure 2. XACML Policy Structure

1) *Deny-overrides*: combines the request evaluation result in such a way that if any rule or a policy evaluated to deny, then the request is denied.
2) *Permit-overrides*: combines decisions such that if any rule or a policy evaluates permission, then the decision is permitted.
3) *First-applicable*: combines decisions in such a way that the final decision is made based on the first rule or policy in the policy file.
4) *Only-one-applicable*: This combining algorithm exists only to combine policy sets and policies. It cannot be used to combine rules. It returns the effect of the unique policy in the policy set that applies to the request; whether Deny or Permit.

Based on the combining algorithm used, PDP computes the authorization decision corresponding to the given access request. PDP evaluation is based on the rule, policy and policy set, for which the PDP returns the authorization decision, *permit*, *deny*, *notApplicable* or *indeterminate*. PDP can also returns to PEP a sequence of actions called *obligation* that should be performed in conjunction with enforcing the authorization decision applied to the access request given.

B. An Example Scenario of Collaboration and Sharing of Healthcare Data

Our implementation is modeled based on a typical user case scenario adopted from [20] and shown in Fig. 3. A patient named *Alice* is recently diagnosed with gastric cancer. Surgical removal of the stomach (gastrectomy) is the only curative treatment. For many patients, chemotherapy and radiation therapy are given after surgery to improve the chances of a cure. *Alice* enters a cancer-treatment center at her chosen hospital (e.g., hospital A). *Alice* has a general practitioner (*Dean*) whom she regularly visits. Upon her hospital visit, *Alice* also sees an attending doctor (*Bob*) from the same hospital. *Alice*'s health condition has caused some complications, so her attending doctor would like to seek expert opinions and consult about *Alice*'s treatment with different hospitals (e.g., hospital B), including *Alice*'s specific general practitioner who is fully informed about *Alice*'s medical history. Note that the invited practitioners are specialized in different areas, where some are specialists and others are general practitioners.

In such group consultation, it is noted that:

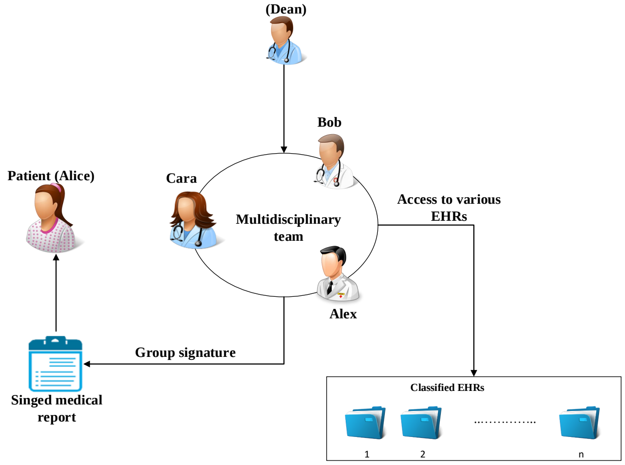


Figure 3. Example scenario of collaboration and sharing of healthcare data

- Several healthcare professionals from different disciplines are involved in various roles to provide patient care.
- The care team are formed dynamically and can be readily changed. For example, when *Alice*'s health condition causes some complications, her attending doctor wishes to seek expert opinions and consult with specialists. As a result of a request for a gastroenterology consultation, we assume a gastroenterologist (*Cara*) will join the care team.
- Every participant needs to obtain some medical records based on the health insurance portability and accountability act (HIPAA) [21] minimal disclosure principle [20], [22].
- Sharing and accessing healthcare records with efficient coordination between healthcare practitioners to perform collaboratively is a critical function in access control models [23]. The main concern regards losing control of sensitive healthcare records while sharing them with multiple parties.

The act of managing the collaborative work in a given scenario must be defined clearly. By default, only the main practitioner (*Dean*) should be aware of the patient's personal information. The three other medical practitioners with supporting roles receive information based on their contributing roles (need-to-know principle). The act of managing a particular collaborative work and how to strike a balance between collaboration and safeguarding sensitive patient information were explained in more detail in our previous works [12], [14], [13].

III. WBAC MODEL IN XACML

The WBAC work model (Fig. 4) [12], [13], [14] postulates that the entire nature of collaboration can be centralized by the *work* concept. Here, each *work* is uniquely identified [13] and is connected to three main components: *personnel*, *patient* and *resources*.

Managing the access control of collaborative work is an interplay between these components. Every resource in WBAC is considered a collaborative entity when it is assigned a *workID*. The *workID* connects the resource to a corresponding work or project that is done cooperatively. By default, if a resource does not have a *workID*, it implies that it is not a collaborative resource and thus cannot be shared.

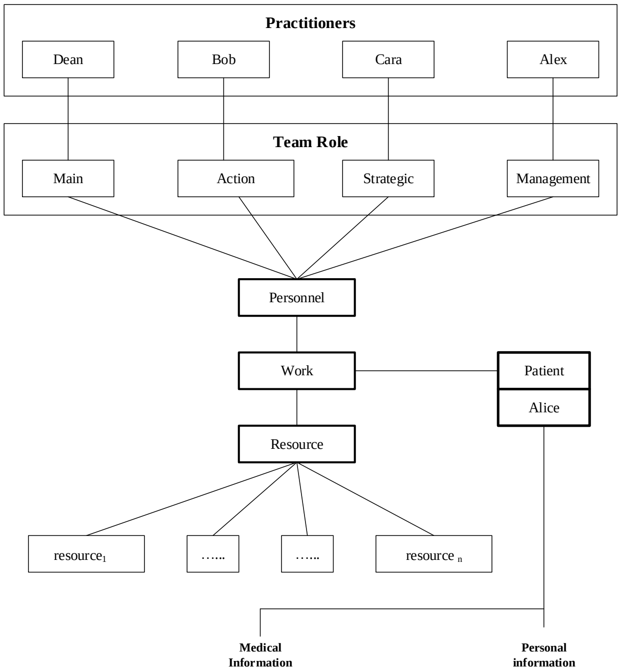


Figure 4. WBAC Work model [14]

Any action that a subject (e.g., healthcare provider) would do on a resource (e.g., patient EHR) is defined entirely within the policy. A dynamic policy with dual inclination is proposed in WBAC [12], [13], whereby the normal policy of enforcing access control is contained within the main policy. On the other hand, any policy that mediates between resource sharing and collaborative work is covered by the collaboration policy. This way, better access control management is achievable. The main policy depends on the roles of the personnel in the organization (e.g., Dean is a general practitioner). PDP only considers the main policy if the personnel possess roles. The collaborative policy is dependent on team roles. In this respect, even if personnel do not have the required roles, they can still gain access upon invitation to collaborate. The team role provides a demarcation between the roles of personnel within a collaboration work and it restricts the role that each team member can have. A person can have various team roles, whereby each is tied to a different collaborative work.

A. Modeling Structures

With the WBAC model, the policy is defined as a tree structure that narrows the combination of attributes presented in a request. Access to a specific resource is granted when the whole policy tree has found possible matches to the request; the result from rule evaluation is then combined upwards to the outer-most policy using the combining algorithm defined at that level. The result is then sent back to the PEP.

The XACML structure of our model is as follows:

1) Subjects, resources and actions are elements defined by identifier/value pairs. Subjects (e.g., healthcare

providers) are entities that send an access request to perform an action (e.g., read or write) on a resource (patient EHRs). The subject is modeled based on the minimal number of attributes required to make different decisions the policy is built to handle. Examples of identifiers are *role*, *employeeID*, *hospitalID* and/or *patientID* (e.g., a patient for whom the physician is responsible), to name a few. For the collaborative part, the information about the subject also includes the team identifier for the current collaboration work. As shown in Fig. 5, physician *Bob* has been assigned the role of attending doctor in the hospital to perform some tasks. He is invited to a collaborative work (*work No 1*) and is assigned the team role *action* to perform some tasks in *Alice*'s treatment.

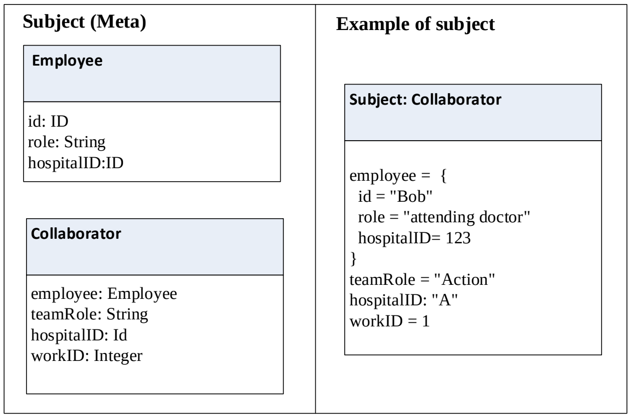


Figure 5. Example of subject attributes

2) *Collaboration members* comprise a group of healthcare providers (specialists or general practitioners) who are invited to a collaborative work (in our case *Alice*'s treatment). Based on the given scenario, *Dean* is responsible for initiating the work and choosing the practitioners (team of doctors) who may be required to attend *Alice*'s consultation and treatment.
Bob, *Cara* and *Alex* joined the team and are assigned team roles based on the required job functions. Table I presents the policy data used as an input for XACML. An action represents the operation that a subject can perform on a resource, e.g., *read* and *write* operations. In our model, we also consider several resource attribute as show in Fig. 6. We also assume the resource are classified into two categories *private* and *protected* (more details about object classification can be found in [13]).

B. Authorization Constraints

We describe the authorization constraints based on the team role classification done in [12], [13], [14] and our usage-scenario (Section II-B) as follows:

- The subject (healthcare provider) who is assigned the primary-doctor role can access both *private* and *protected* resources of the patient for whom he/she is responsible.
- A collaborative work must be active, such that team members can work on it. Assuming the value set assigned to a work is its identifier, and if there is no work, the field will not be present in a request.

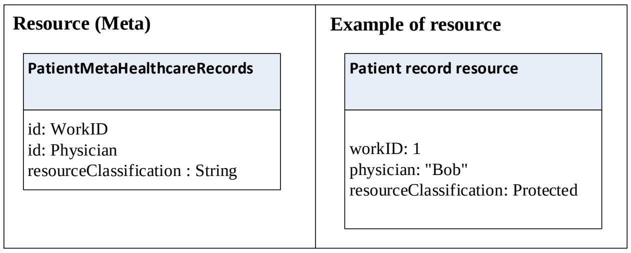


Figure 6. Example of resource attributes

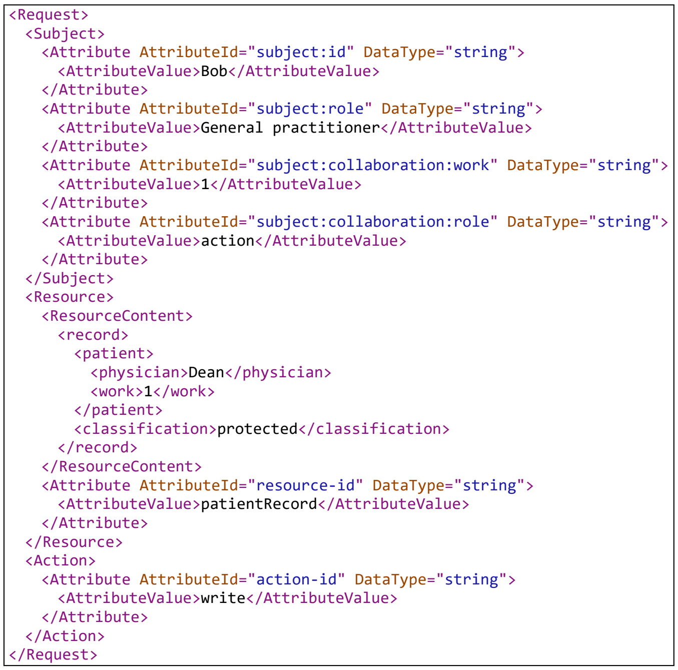
- Only a subject (healthcare providers) who is a member of the care team and is assigned the *action* team role can access *private* and *protected* resources, but only if needed (inevitably). In this model, we assume the healthcare provider who is assigned the *action* team role needs to access private resources because he/she needs to see a patient on a face-to-face basis to perform various tasks related to the patient's recovery. In this respect, there is a need for the healthcare provider to know personal and medical information about the patient to perform his/her duty effectively. Note that in other scenarios, a healthcare provider who is assigned the *action* team role might not need to know private information about the patient.
- Only a subject (healthcare providers) who is a member of the care team and who is assigned the *strategic* team role can access *protected* resources, which are approved for collaboration works. This healthcare provider is predominantly preoccupied with diagnosing the disease, and there is no urgent need for him/her to know the patient's personal information. He/she is responsible for helping the primary doctor to solve the medical case. In fact, he/she is only required to analyze the medical situation and suggest a possible solution. In our model, personnel assigned the *strategic* team role are permitted access only to *protected* resources (e.g., any resources related to the current case of the patient).
- Healthcare providers who are assigned the *management* team role are responsible for coordinating the other team members' interaction by managing meetings and resolving problems with conflicting diagnoses made by other team members. The healthcare provider does not really need to know the patient's personal information. However, he/she must be aware of the patient's medical information to enable coordination. Similar to the *strategic* team role, personnel assigned the *management* team role are permitted access only to *protected* resources. The difference between the *strategic* and *management* team roles is the need for personnel assigned to the *management* team role to have access to team member (healthcare provider) records to be informed of specialist information related to the team members (physicians) in order to coordinate the collaborative work effectively.

TABLE I. Tabular structure of policy data

Subject	Job Function	Team Role	Object Type	Action	Permission
Dean	Primary Doctor	Main role	Private and protected	Read/write	Permit
Bob	General practitioner	Action	Private and protected	Read	Permit
Cara	Gastroenterologist	Strategic	Protected	Read	Permit
Alex	Medical coordinator	Management	Protected	Read	Permit

C. Request Model

The XACML request contains the attributes related to subject, resource and action with their corresponding values. For example, in our case and as depicted in Fig. 7 we have attribute *Subject:Role* and its value *General practitioner*, and attribute *ResourceClassification* and its value *protected* as well as an action value *write*. This information is necessary for authorization decision-making. When PDP evaluates the request against the policy, the attribute names and attribute values are compared according to criteria defined in the policy.



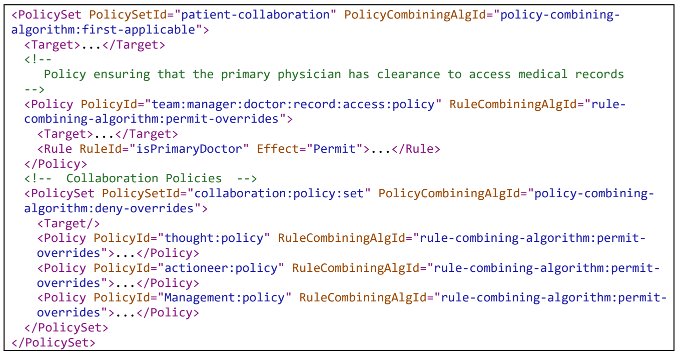
```
<Request>
  <Subject>
    <Attribute AttributeId="subject:id" DataType="string">
      <AttributeValue>Bob</AttributeValue>
    </Attribute>
    <Attribute AttributeId="subject:role" DataType="string">
      <AttributeValue>General practitioner</AttributeValue>
    </Attribute>
    <Attribute AttributeId="subject:collaboration:work" DataType="string">
      <AttributeValue>1</AttributeValue>
    </Attribute>
    <Attribute AttributeId="subject:collaboration:role" DataType="string">
      <AttributeValue>action</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <ResourceContent>
      <record>
        <patient>
          <physician>Dean</physician>
          <work>1</work>
        </patient>
        <classification>protected</classification>
      </record>
    </ResourceContent>
    <Attribute AttributeId="resource-id" DataType="string">
      <AttributeValue>patientRecord</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="action-id" DataType="string">
      <AttributeValue>write</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Figure 7. Example of an XACML access request

D. Policy Model

The XACML collaboration model begins with a top-level policy set containing one policy for handling a case where the subject is the patient's primary physician and a policy set for the different collaboration cases as shown in Fig. 8.

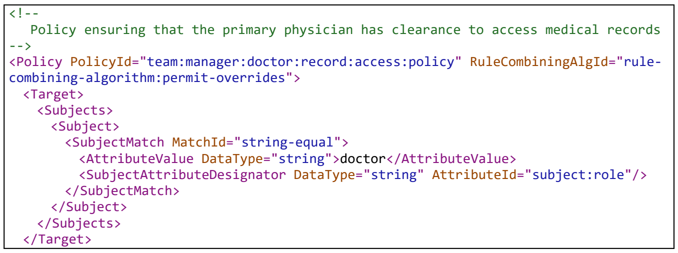
The top-level policy combines the results based on first applicability, meaning that if the requesting subject is the patient's primary doctor, he/she will get access to records regardless of collaboration. PDP will receive all policies as inputs, where each policy has an element known as ”target”. The target element's attribute values (subject, resource, action and environment) are matched with the incoming request (Fig. 7) attribute values to decide whether a particular policy is applicable to a given request. If the request attributes match the target's attributes, the policy will be evaluated further. Else, PDP decides the given request is not applicable to the policy.



```
<PolicySet PolicySetId="patient-collaboration" PolicyCombiningAlgId="policy-combining-
algorithm:first-applicable">
  <Target>...</Target>
  <!--
     Policy ensuring that the primary physician has clearance to access medical records
  -->
  <Policy PolicyId="team:manager:doctor:record:access:policy" RuleCombiningAlgId="rule-
  combining-algorithm:permit-overrides">
    <Target>...</Target>
    <Rule RuleId="isPrimaryDoctor" Effect="Permit">...</Rule>
  </Policy>
  <!--  Collaboration Policies  -->
  <PolicySet PolicySetId="collaboration:policy:set" PolicyCombiningAlgId="policy-combining-
  algorithm:deny-overrides">
    <Target/>
    <Policy PolicyId="thought:policy" RuleCombiningAlgId="rule-combining-algorithm:permit-
    overrides">...</Policy>
    <Policy PolicyId="actioneer:policy" RuleCombiningAlgId="rule-combining-algorithm:permit-
    overrides">...</Policy>
    <Policy PolicyId="Management:policy" RuleCombiningAlgId="rule-combining-algorithm:permit-
    overrides">...</Policy>
  </PolicySet>
</PolicySet>
```

Figure 8. Screenshot of top-level policy set

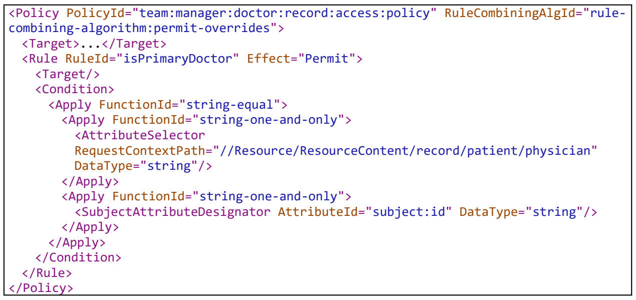
Fig. 9 displays a sample policy (part of the defined policy), which ensures the primary physician has clearance to access medical records.



```
<!--
   Policy ensuring that the primary physician has clearance to access medical records
-->
<Policy PolicyId="team:manager:doctor:record:access:policy" RuleCombiningAlgId="rule-
combining-algorithm:permit-overrides">
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId="string-equal">
          <AttributeValue DataType="string">doctor</AttributeValue>
          <SubjectAttributeDesignator DataType="string" AttributeId="subject:role"/>
        </SubjectMatch>
      </Subject>
    </Subjects>
  </Target>
```

Figure 9. Example of a policy ensuring that the primary physician has clearance to access medical records

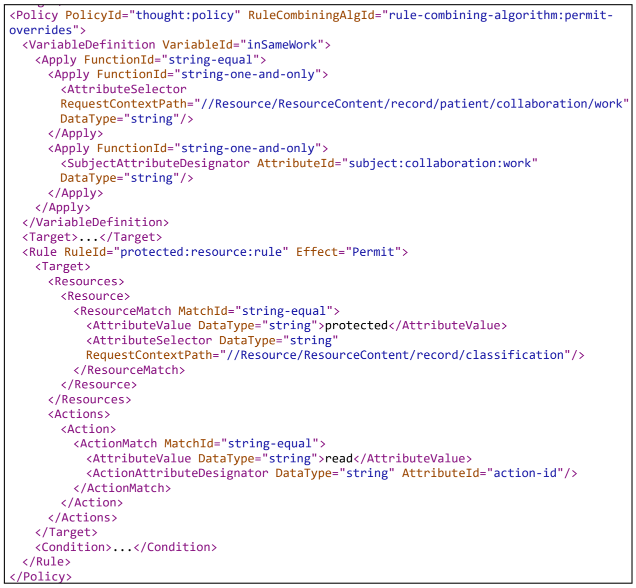
While the target element evaluates the applicability of a policy, the rule element implements the actual authorization logic. The primary physician policy has one rule as demonstrated in Fig. 10, which permits access. If the rule's condition is evaluated as true, the output of the rule will be ”permit” where the primary physician field in the resource content patient metadata the same identifier for the subject.



```
<Policy PolicyId="team:manager:doctor:record:access:policy" RuleCombiningAlgId="rule-
combining-algorithm:permit-overrides">
  <Target>...</Target>
  <Rule RuleId="isPrimaryDoctor" Effect="Permit">
    <Target/>
    <Condition>
      <Apply FunctionId="string-equal">
        <Apply FunctionId="string-one-and-only">
          <AttributeSelector
          RequestContextPath="//Resource/ResourceContent/record/patient/physician"
          DataType="string"/>
        </Apply>
        <Apply FunctionId="string-one-and-only">
          <SubjectAttributeDesignator AttributeId="subject:id" DataType="string"/>
        </Apply>
      </Apply>
    </Condition>
  </Rule>
</Policy>
```

Figure 10. Example of a rule that defines the primary physician is permitted access to medical records

Collaboration policies are divided into three sub-policy sets from the main policy set, as shown in Fig. 8. Each policy set is for one specific team role and the rule that applies to this team role. To evaluate collaborative work, the subject *workID* is matched with that of the resource and must be equal for access to be granted and combined with other constraints, such as *read* or *write* effect. An instance of one collaboration policy is shown in Fig. 11. Here, the subject assigned the *strategic* team role is granted access (read access only) to the *protected* resource type if the *workID* matches the active *workID*.



```
<Policy PolicyId="thought:policy" RuleCombiningAlgId="rule-combining-algorithm:permit-
overrides">
  <VariableDefinition VariableId="inSameWork">
    <Apply FunctionId="string-equal">
      <Apply FunctionId="string-one-and-only">
        <AttributeSelector
        RequestContextPath="//Resource/ResourceContent/record/patient/collaboration/work"
        DataType="string"/>
      </Apply>
      <Apply FunctionId="string-one-and-only">
        <SubjectAttributeDesignator AttributeId="subject:collaboration:work"
        DataType="string"/>
      </Apply>
    </Apply>
  </VariableDefinition>
  <Target>...</Target>
  <Rule RuleId="protected:resource:rule" Effect="Permit">
    <Target>
      <Resources>
        <Resource>
          <ResourceMatch MatchId="string-equal">
            <AttributeValue DataType="string">protected</AttributeValue>
            <AttributeSelector DataType="string"
            RequestContextPath="//Resource/ResourceContent/record/classification"/>
          </ResourceMatch>
        </Resource>
      </Resources>
      <Actions>
        <Action>
          <ActionMatch MatchId="string-equal">
            <AttributeValue DataType="string">read</AttributeValue>
            <ActionAttributeDesignator DataType="string" AttributeId="action-id"/>
          </ActionMatch>
        </Action>
      </Actions>
    </Target>
    <Condition>...</Condition>
  </Rule>
</Policy>
```

Figure 11. An example of one of the collaboration policy (`strategic` team role policy)

IV. Experiments and Result

The WBAC model as described in Section III has been implemented using XACML 2.0. Verifying that this implementation of WBAC can be used as part of an XACML policy was done using the Java SunXACML implementation [24] to run a PDP, testing the policy against different requests. The experiment assumes that the PDP is configured to be deny-biased which means that any response which is *indeterminate* or *not applicable* is seen as a *deny* response.

The WBAC policy was tested by using the attributes based on the data models shown in Fig. 5 and Fig. 6 to build access control requests as shown in Fig. 7. Both valid and invalid values were set for the different attributes to verify that access was permitted and denied correctly.

The experiments showed that the WBAC model granted access correctly to subjects matching the same work as the resource for the expected cases. Invalid request such as a subject *work* with the value 2, while the resource *work* value set to 1. Since the policy is only implemented with rules needed for permitting access when requests is matched the PDP responded with a *indeterminate* answer, which is interpreted as a deny response when the PDP is deny biased.

V. Discussion and Conclusions

A. Related Works

Researchers have made the best effort to propose an access control model that balance between security and collaboration requirements [25], [26], [27]. A numerous of research trends on access control approaches have been presented such as RBAC, ABAC, team-based access control (TMAC) [15], task-based access control (TBAC) [28], context-based TMAC (C-TMAC) [16], team task based RBAC (TT-RBAC) [29] and group-based RBAC (GB-RBAC) [30]. In this Section, we compare them to understand better the differences between these approaches. Comparison is imperative and aims at well defining the appropriate access control model for our model. The main evaluation criteria for access control in collaborative system were presented in number of studies [26], [31]. The assessment criteria with respect to healthcare collaborative environments as follows:

1) **Personalized permission**: Patients must be informed of the collaboration and should be given the right to choose who can have access to their records.
2) **Selective confidentiality**: Certain patient information is highly sensitive. Thus, patients should be able to withhold information that remains confidential.
3) **Flexibility and adaptability**: Flexibility is the ability of the access control model to support frequent changes in policy. Whereas, adaptability is used to evaluate the ability of the access control to adapt different healthcare scenarios and environments.
4) **Fine-grained control**: the access control model should support fine-grained subjects, objects and access rights. The granular level at which rules can be applied not only for roles but also for individuals on one or many controlled objects [26].
5) **Groups of users: assignment and revocation**: in collaborative work, common tasks are undertaken by a group of people (team). Therefore, access control model supports the notion of team and allows to specify access rights for teams. Also, the model should has the capability to revoke rights of subjects to access objects.
6) **Policy specifications**: The access control model should allows for scalability and easy extension and modifications of access rights of subjects to access objects.
7) **Policy enforcement**: The access control model should provide means that ensures a correct enforcement of the policy or constraints specification.
8) **Designed for collaborative Healthcare systems**: This criteria show whether or not the access control solutions was designed specifically for collaborative Healthcare systems

Table II summarizes our comparative analysis of the RBAC, ABAC, TMAC, TBAC, C-TMAC, TT-RBAC, GB-RBAC and WBAC models to access control. The table make use of the comparative terminology where ”Low”, ”Medium”, and ”High” are used to indicate the degree to which the requirement is supported. Also, descriptive terminology such as ” Complex” is used to describe the level to which the requirement is supported as well as the standard terminology ”Yes”, and ”No” have been used whenever it is possible to

TABLE II. Access control methods comparison

Access Control models	**Assessment Criteria**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
RBAC	No	No	Medium	Low	Yes	Yes	Yes	Yes
ABCA	No	Yes	High	High	Complex	Complex	Complex	No
TMAC	Yes	Medium	Medium	Yes	Medium	Yes	Yes	No
TBAC	No	Medium	Low	Low	No	Yes	Low	No
C-TMAC	Yes	No	Medium	Yes	Yes	Yes	Yes	No
TT-RBAC	Yes	No	High	Yes	Yes	Complex	Yes	Yes
GB-RBAC	Yes	No	Medium	Low	Yes	Yes	Yes	No
WBAC	Yes	Yes	High	Medium	Yes	Yes	Yes	Yes

indicate the facilitation or lack of facilitation of the concerned the requirement by the access control model.

WBAC is loosely based on RBAC and ABAC models, and extended with a team role concept. Role and team role are used in conjunction to deal with access control in dynamic collaborative environments. Therefore, the flexibility and adaptability, fine-grained control, policy specifications and policy enforcement are the same as the RBAC. Groups of users: assignment and revocation is similar to TMAC, C-TMAC and TT-RBAC except that in WBAC, the team is classified based on the team role . In WBAC, one team can assigned to a collaborative work in any granularity based on team members' team role. Selective confidentiality is well supported by WBAC because, it is possible to assign a specific object to a each team member in given team based on the object classification and team role. we believe that WBAC handles personalized permission well and meets our expectation of allowing fine-grained access control as well as enhance the practicability and manageability of access control in dynamic collaboration environment. That is, we assume that WBAC is at least as good as RBAC and ABAC, model in this area.

B. Discussion

To prevent any violation of the access control policy of an organization, most classical access control models like RBAC and ABAC define users rights precisely, based on subject and object elements. When several subjects and objects are involved, the subject-object model cannot deliver satisfactory security management. In collaborative environments such as healthcare, it is challenging to predefine all access needs based on the subject-object model. One example of such a situation is explained in our case scenario (Section II-B), which may not be predictable and it would be hard to express the condition of who should join the collaboration and when *Dean* necessitates collaborative support from other parties. Moreover, in deciding on the extent and limit of resource sharing, For instance, in the case of *Alice*'s treatment, which sensitive data should be disclosed to an assisting practitioner so collaboration can be effective, and which should be hidden to safeguard the patient's privacy? Another important matter is the correctness of the policy. Access policy adoption may be limited if the intended policies are not implemented efficiently and consequently thus perform poorly.

WBAC was proposed to address these concerns and support the security and collaboration requirements in access control [25], [26], [27]. The major contributions of the WBAC model include ensuring that access rights are dynamically adapted to the actual needs of healthcare providers and providing fine-grained control of access rights with the least privilege principle, whereby healthcare providers are granted minimal access rights to carry out their duties. In our case scenario, it was noted that general practitioner *Dean* could not solve Alice's case alone. He invited a multidisciplinary team including *Bob*, *Cara* and *Alex* to help. In this team, *Dean* is the core physician in the collaborative work and servers as the group manager. He is responsible for initiating the *work* (*Alice*'s treatment case) and choosing practitioners (group of doctors) who may be required to attend *Alice*'s consultation and treatment. This implies that *Dean* holds the main role. In other words, he owns the initiated collaborative work. Therefore, *Dean* is given a full access (based on his role as primary physician, Fig. 9) with regard to patient-related information. *Bob*, *Cara* and *Alex* are assigned to team roles based on the job function they will perform in *Alice*'s treatment. In our previous work [13], we formally describe and showed how each user joins the team and how each should be assigned at least one team role; a team role can be assigned to none or multiple users in many teams.

In this study, we demonstrated WBAC policies in XACML. We selected XACML because it has been proven to be adaptable to specifying several common access control methods, such as RBAC and ABAC. Our implementation only covers access request for medical records resources, but by using similar matching technique as for the *work* attribute, it is possible to extend this to other polices that are also active during collaboration. An example of this could be for persons with the $management$ team role, which should also have access to the personal files like those in the same collaborative work team.

XACML offers extensibility and pluggability which enables the policy presented in this work to be not only a standalone policy, but it could also be a small part of a larger collection of policies. Possible extensions of the base collaboration policy could, for example, be sub-roles of each primary collaboration roles. This could give even more granularity for specific cases for example if a medical employee in the $management$ team role.

C. Conclusions and Future Work

A work-based access control (WBAC) model was proposed, which is suitable for collaborative healthcare systems in addressing the subjects of information sharing and security. The aim was to provide a flexible access control model without compromising the granularity of access rights. In this study, we showed how XACML can be used to implement the WBAC model policy and how XACML combining algorithms can be used to manage the inconsistencies between different policy sets. XACML has become very popular in both academia and industry as a standard for combining, maintaining and exchanging access control policies. It is an architecture for evaluating authorization requests and for issuing authorization decisions. The experiments we conducted demonstrated the applicability of XACML to supporting collaborative and distributed domains in sharing access control of specific resources.

In the future, the plan is to formalize access control policies and use automated verification tools to verify interesting properties about the WBAC policy as well as detect consistency using such automated tools. SAT Solver [32] and Alloy [33] are examples of automated verification tools. The plan is also to prototype the functionality to be implemented, to ensure the model's practicality and evaluate the validity of the possible difficulties in managing the model during actual implementation.

REFERENCES

[1] P. Samarati and S. D. C. Di Vimercati, "Access control: Policies, models, and mechanisms," Lecture notes in computer science, 2001, pp. 137–196.

[2] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed nist standard for role-based access control," ACM Transactions on Information and System Security (TISSEC), vol. 4, no. 3, 2001, pp. 224–274.

[3] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (abac) definition and considerations," NIST Special Publication, vol. 800, 2014, p. 162.

[4] M. H. Kang, J. S. Park, and J. N. Froscher, "Access control mechanisms for inter-organizational workflow," in Proceedings of the sixth ACM symposium on Access control models and technologies. ACM, 2001, pp. 66–74.

[5] S. Oh and S. Park, "Task–role-based access control model," Information systems, vol. 28, no. 6, 2003, pp. 533–562.

[6] R. A. Shaikh, K. Adi, L. Logrippo, and S. Mankovski, "Inconsistency detection method for access control policies," in Information Assurance and Security (IAS), 2010 Sixth International Conference on. IEEE, 2010, pp. 204–209.

[7] Q. Li, X. Zhang, M. Xu, and J. Wu, "Towards secure dynamic collaborations with group-based rbac model," computers & security, vol. 28, no. 5, 2009, pp. 260–275.

[8] D. Daiqin He and J. Yang, "Authorization control in collaborative healthcare systems," Journal of theoretical and applied electronic commerce research, vol. 4, no. 2, 2009, pp. 88–109.

[9] H. Janicke, A. Cau, F. Siewe, and H. Zedan, "Dynamic access control policies: Specification and verification," The Computer Journal, 2012, p. bxs102.

[10] G. Zhang and M. Parashar, "Context-aware dynamic access control for pervasive applications," in Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference, 2004, pp. 21–30.

[11] T. Moses et al., "Extensible access control markup language (xacml) version 2.0," Oasis Standard, vol. 200502, 2005.

[12] M. Abomhara and G. M. Køien, "Towards an access control model for collaborative healthcare systems," in Proceedings of the 9th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2016), vol. 5, 2016, pp. 213–222.

[13] M. Abomhara, H. Yang, and G. M. Køien, "Access control model for cooperative healthcare environments: Modeling and verification," in IEEE International Conference on Healthcare Informatics 2016 (ICHI 2016) (accepted), 2016.

[14] M. Abomhara and H. Yang, "Attribute-based authenticated access for secure sharing of healthcare records in collaborative environments," in Proceedings of the The Eighth International Conference on eHealth, Telemedicine, and Social Medicine 2016 (eTELEMED), 2016.

[15] R. K. Thomas, "Team-based access control (tmac): a primitive for applying role-based access controls in collaborative environments," in Proceedings of the second ACM workshop on Role-based access control. ACM, 1997, pp. 13–19.

[16] C. K. Georgiadis, I. Mavridis, G. Pangalos, and R. K. Thomas, "Flexible team-based access control using contexts," in Proceedings of the sixth ACM symposium on Access control models and technologies. ACM, 2001, pp. 21–27.

[17] J. F. Alqatawna, E. Rissanen, and B. Sadighi, "Overriding of access control in xacml," in Policies for Distributed Systems and Networks, 2007. POLICY'07. Eighth IEEE International Workshop on. IEEE, 2007, pp. 87–95.

[18] A. X. Liu, F. Chen, J. Hwang, and T. Xie, "Designing fast and scalable xacml policy evaluation engines," Computers, IEEE Transactions on, vol. 60, no. 12, 2011, pp. 1802–1817.

[19] N. Li, Q. Wang, W. Qardaji, E. Bertino, P. Rao, J. Lobo, and D. Lin, "Access control policy combining: theory meets practice," in Proceedings of the 14th ACM symposium on Access control models and technologies. ACM, 2009, pp. 135–144.

[20] R. Zhang and L. Liu, "Security models and requirements for healthcare application clouds," in Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. IEEE, 2010, pp. 268–275.

[21] S. J. Dwyer III, A. C. Weaver, and K. K. Hughes, "Health insurance portability and accountability act," Security Issues in the Digital Medical Enterprise, vol. 72, no. 2, 2004, pp. 9–18.

[22] B. Fabian, T. Ermakova, and P. Junghanns, "Collaborative and secure sharing of healthcare data in multi-clouds," Information Systems, vol. 48, 2015, pp. 132–150.

[23] F. T. Alotaiby and J. X. Chen, "A model for team-based access control (tmac 2004)," in Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on, vol. 1. IEEE, 2004, pp. 450–454.

[24] S. Microsystems, "Official project web site sun's xacml implementation," "[accessed 23-May-2016]. [Online]. Available: http://www.sunxacml.sourceforge.net

[25] B. Alhaqbani and C. Fidge, "Access control requirements for processing electronic health records," in Business Process Management Workshops. Springer, 2008, pp. 371–382.

[26] W. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong, "Access control in collaborative systems," ACM Computing Surveys (CSUR), vol. 37, no. 1, 2005, pp. 29–41.

[27] H. Shen and P. Dewan, "Access control for collaborative environments," in Proceedings of the 1992 ACM conference on Computer-supported cooperative work. ACM, 1992, pp. 51–58.

[28] R. K. Thomas and R. S. Sandhu, "Task-based authorization controls (tbac): A family of models for active and enterprise-oriented authorization management," in Database Security XI. Springer, 1998, pp. 166–181.

[29] W. Zhou and C. Meinel, "Team and task based rbac access control model," in Network Operations and Management Symposium, 2007. LANOMS 2007. Latin American. IEEE, 2007, pp. 84–94.

[30] Q. Li, M. Xu, and X. Zhang, "Towards a group-based rbac model and decentralized user-role administration," in 2008 The 28th International Conference on Distributed Computing Systems Workshops. IEEE, 2008, pp. 441–446.

[31] S. Alshehri and R. K. Raj, "Secure access control for health information sharing systems," in Healthcare Informatics (ICHI), 2013 IEEE International Conference on. IEEE, 2013, pp. 277–286.

[32] G. Hughes and T. Bultan, "Automated verification of xacml policies using a sat solver," in Proceedings of the Workshop on Web Quality, Verification and Validation (WQVV07), 2007.

[33] X. Wang and A. Rutle, "Model checking healthcare workflows using alloy," Procedia Computer Science, vol. 37, 2014, pp. 481–488.