A Study of Attribute Based Access Control Policies with Ontology

Rukmangathan

Supervisor: Dr. Bernard Butler
Course: MSc. in Computing (Enterprise Software Systems)

Table of Contents

1 List of abbreviations

RBAC:	**Role Based Access Control**
ABAC:	**Attribute Based Access Control**
XACML:	**Extensible Access Control Markup Language**
PDP:	**Policy Decision Point**
OWL:	**Web Ontology Language**
PEP:	**Policy Evaluation Point**
PIP:	**Policy Information Point**
PAP:	**Policy Access Point**

2 Plagiarism Declaration

I certify that this assignment is all my own work and contains no Plagiarism. By submitting this assignment, I agree to the following terms:

Any text, diagrams or other material copied from other sources (including, but not limited to, books, journals and the internet) have been clearly acknowledged and referenced as such in the text using quotation marks (or indented italics for longer quotations) followed by the authors name and date either in the text or in a footnote/endnote. These details are then confirmed by a fuller reference in the bibliography.

I have read the sections on referencing and plagiarism in the handbook or in the SETU Plagiarism policy and I understand that only assignments which are free of plagiarism will be awarded marks. I further understand that SETU has a plagiarism policy which can lead to the suspension or permanent expulsion of students in serious cases.

3 Introduction

Web security can be referred as the protective measures an application owner, or an organization takes to safeguard data from cyber-criminal and any other form of threat which may lead to data compromise or unauthorized access. Securing the web application from malicious threats is becoming a harder task every day. Securing the infrastructure is essential for any organizations or business for a smooth operation.

The more formal definition of web security according to Mozilla (2021) is “The purpose of website security is to prevent these (or any) sorts of attacks. The more formal definition of website security is the act/practice of protecting websites from unauthorized access, use, modification, destruction, or disruption”

According to cyber security statistics (2021) 95% of the security compromises are caused by human error.

There are a variety of security schemes and systems in place to defend the web system from various attacks. One such area is the access management which leverages access control systems to restrict unauthorized or unintended access to sensitive digital assets. These systems bring immense security to value to any organization as businesses lose millions of euros every year which can be prevented if access control systems are in place.

Attribute based access control (ABAC) and Role based access control (RBAC) are two of the popular ways to implement access control.

In RBAC permissions are assigned to a user based on their role in the organization. Generally, while implementing RBAC, the users are group into different roles based on their responsibilities. A user can be assigned one or more role and their access to different system resources if based on the roles that they are assigned to.

For example, If RBAC was used to control access for a hospital data management software, The chief doctor can be given chief role that allows them to view all patient details, while other doctors would be able to view only their patient details.

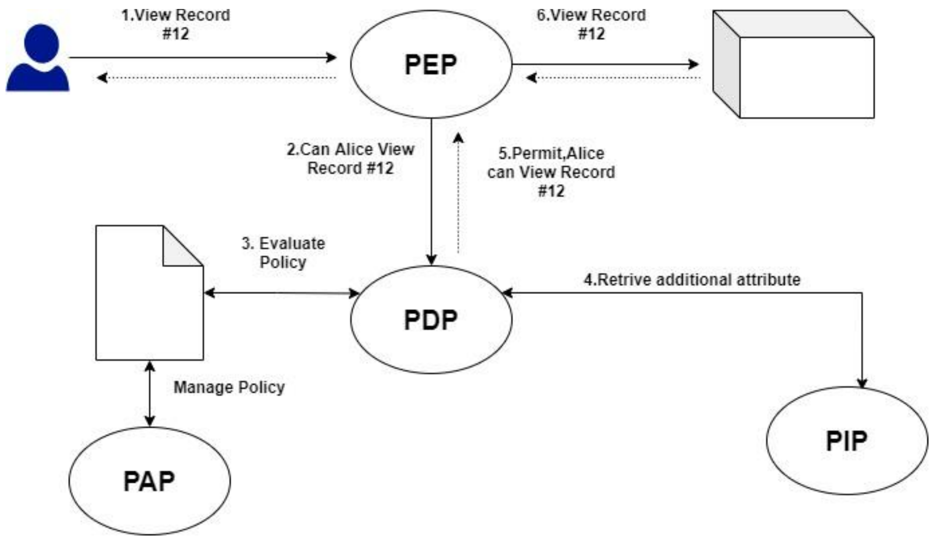
Unlike RBAC which uses roles to determine the permissions for a user, The Attribute Based Access Control uses a set of attributes that are assigned to a user to evaluate it against a set of prewritten policies to determine whether a user is allowed access to a resource or to perform the desired operation. The policy evaluation engine or Policy Decision Point (PDP) takes care of evaluating a requested user's attributes with the defined policies. The PDP can be thought of as the brain of the ABAC system.

The ABAC provides fine grained access control compared to RBAC in a way that a user's operation on a resource can be restricted based on attributes like the time of the day, the environment the request came from or the type of operation the user is trying to perform.

While RBAC works very well for smaller work groups and provides a coarse grain access control, for more complex scenarios, ABAC is generally preferred as it easily scalable, provides much more granular access control and it is often referred to as an evolved form of RBAC.

The policies in ABAC are a set of statements that can bring together attributes to expression what is allowed and what is denied. The way of defining policies differ based on the policy language used. While there are several policy language and implementation model available for ABAC, the eXtensible Access Control Markup Language (XACML) is the most prevalent and widely used policy language. The Extensible Access Control Markup Language (XACML) is an OASIS standard which is currently in version 3.0. XACML is an attribute-based policy language that was designed to help with interoperability in authorization across applications. XACML is a standardized Policy evaluation model that abstracts the major features of modern enterprise access control systems.

3.1 ABAC architecture / XACML


1.View Record #12
PEP
6.View Record #12
2.Can Alice View Record #12
5.Permit,Alice can View Record #12
3. Evaluate Policy
PDP
4.Retrive additional attribute
Manage Policy
PIP
PAP

(Source: elixirforum 2022)

3.2 Sample XACML policies

Sample XACML request

```
<Request
xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"  xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" >
  <Subject>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="xs:string" Issuer="med.example.com">
      <AttributeValue>CN=Homer Simpson</AttributeValue>
    </Attribute>
</Subject>
  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#anyURI"/>
      <AttributeValue>https://med.example.com/BartSimpsonRecord.jws</AttributeVa
lue>
    </Attribute>
</Resource>
  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="xs:string">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
  <Environment/>
</Request>
```

Sample XACML response

```
<xacml-ctx:Response xmlns:xacml-
ctx="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17">
  <xacml-ctx:Result>
    <xacml-ctx:Decision>NotApplicable</xacml-ctx:Decision>
    <xacml-ctx:Status>
      <xacml-ctx:StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </xacml-ctx:Status>
  </xacml-ctx:Result>
</xacml-ctx:Response>
```

The XACML policy language has several pitfalls that makes it hard to use. Some of the key challenges when working with XACML based policies are:

1. As organizations grow and evolve, the XACML polices can become verbose and very complex to write and maintain.
2. It generally requires a technical person to write and maintain XACML policies.
3. Interpretation of complex policies are difficult.
4. When working with multiple policies across different targets, it can be very easy to make mistake that could cause bugs.

The focus of this research is to make ABAC policy configuration easier to create, configure & manage ABAC policies in a human friendly way and to find ambiguities in the configurations and alert the end user from providing meaningless configurations which would result in devastating security breach and data compromise. The research is geared towards representing ABAC policy data through ontology. Through my preliminary literature review I discovered that no solid research work has been done in this area.

Ontologies are a formal way to describe taxonomies and classification networks, essentially defining the structure of knowledge for various domains: the nouns representing classes of objects and the verbs representing relations between the objects (Wikipedia) Ontologies in general are simple, effective, and easy way of representing the entities and relationships between them.

The overall goal of the research is to make the life of the ABAC policy author easier and lay a foundation for research directed in refinement of policy languages.

4 Motivation

I am a software professional with emphasis on security.

Over the past, I have been a part of two organizations where I worked on the team that used to implement access control. The experience fascinated me on many levels and made me more curious about security in general. Along the journey, I was able to understand how complex an access control system gets when the application keeps evolving. Making changes in the access control system would require careful testing and would burn a lot of time. This was the driving factor which made me realize that there is room for improvements in the access control area and there is a wide array of problems to solve.

Also, the bugs that I've found in access control systems of various web applications while I was doing bug bounty hunting as hobby, fueled my motivation to pursue research work in the ABAC area.

One of the popular crowd sourced bug bounty program(Hacker one 2021) reports a 53% percent surge in bugs that are caused due to improper access control in the year 2021. The improper access control is mainly caused by human errors when writing policies. As policies get more complex as the system evolves, the new changes tend to break the existing changes thereby causing a bug in the system.

With the above statistical data coupled with my experience, I strongly believe that the wide range of access control problems are caused to the misconfiguration which can be prevented if the way of writing down policies are intuitive and human friendly.

5 Scope

The scope of this research is focused around XACML language and OWL. The work cannot be generalized to any other policy languages for ABAC. The following areas will be covered.

- Modelling XACML policies in OWL
- Analyzing XACML equivalent policies in OWL and discovering potential problems/ambiguities using entity relationships.

The following areas will not be covered:

- Other policy languages are out of scope.
- To narrow down scope, this research will only be focused on basic policies and complex polices like break glass policies are considered out of scope.

6 Research questions

- What role can ontologies play in managing ABAC policies?
- How can ontologies be leveraged to make ABAC policy author’s life easier?

7 Preliminary literature review

When deployed across an enterprise to increase information sharing among diverse organizations, ABAC implementations can become complex—supported by the existence of an attribute management infrastructure, machine-enforceable policies, and an array of functions that support access decisions and policy enforcement.
(Hu et al., 2015).

XACML is simply an access control policy language, it lacks any kind of formal model of ABAC despite its support for attributes, making it at best only one component of a larger model. Wang et al.’s logic-based framework, provides a start towards a generic foundational model but mostly concentrates on modelling policies and their evaluation and cannot be seen as a complete model of ABAC.
(Osborn & Servos, 2017a)

In this paper we have shown how the *Attribute Based Access Control* model can be represented using *Web Ontology Language* (OWL). This is a starting step towards formally specifying and enforcing machine understandable policies that can be captured in the ABAC model, which is one of the most general access control models available today.
(Sharma & Joshi, 2016)

8 Methodology

A quantitative approach will be used for this research work. This includes but not limited to experimenting and exploring the relationships between different ABAC entities. The research work will make use of the following tools:

8.1 Protege

Protégé is an open-source ontology editor and framework developed by the Stanford Center for Biomedical Informatics Research for building knowledge-based solutions. It provides a plug-and-play environment that makes it a flexible base for rapid prototyping.

8.2 OWL 2.0

The Web Ontology Language (OWL) is a family of knowledge representation languages for authoring ontologies. OWL has three increasingly expressive sublanguages: OWL Lite, OWL DL, and OWL Full. OWL Full provides more support for expressing entities and it will be leveraged for the purpose of this project.

8.3 Python

Python is a feature rich high level programming language which is popular for scripting and writing web applications. In this research we would use python for parsing OWL documents and XML parsers to re-reconstruct ontologies in the form of XACML.

8.4 Policy Decision Point

A PDP will be required to verify and validate the end policies that are generated from ontology representations. Luas, Node js based PDP Supports that XACML 3.0 will be targeted to evaluate and use.

9 Anticipated milestones

- Explore and gain in-depth knowledge on OWL and XACML – Jan 9-21
- Test and evaluate the tools and software picked – Jan 21-31
- Develop test data set and environment creation – Feb 1-11
- Checkpoint1 for making progress on dissertation report – Feb 12-25
- Represent XACML in OWL using Protégé – Feb 26 – Mar 11
- Interim report Submission – Mar 12-31
- OWL to XACML conversion model – Apr 1-15
- Checkpoint2 for making progress on dissertation report – Apr 16-30
- Analysing ABAC policies represented in OWL – May 1-31
- Complete draft of final dissertation for supervisor – June 1-30
- Final report and presentation – Jul 1-30
- Final dissertation presentation & assessment – Aug 1-30

10 Summary

Date range	Anticipated Milestone	Anticipated Tasks	Key deliverables
Jan 9-21	Explore and gain in-depth knowledge on OWL and XACML	Explore OWL documentation and get familiar with syntaxes and relationships	• Should able to design and develop complex ontologies • Should be able to create and test XACML policies
		Explore and gain insights on XACML 3.0	
Jan 21-31	Test and evaluate the tools and software picked	Test XML parsing library	Tools should be ready to use
		Test Python OWL library	
		Test Luas PDP	
Feb 1-11	Develop test data set and environment creation	Come up with a data set that replicates an ABAC scenario	Test environment should be ready
Feb 26 – Mar 11	Represent XACML in OWL using Protégé	Represent an XACML policy in OWL	XACML policy should be represented in OWL as an ontology
Apr 1-15	OWL to XACML conversion model	Prototype that can convert OWL policies into XACML	The prototype should be able to convert a policy defined via OWL to XACML
May 1-31	Analysing ABAC policies represented in OWL	Use the entity relationships created through ontology and analyse the ABAC policy to provide warnings or show information about ambiguity in the policy	There should a prototype ready to show possible warning in the policies.

11 References

(*A Beginner's Guide to XACML. XACML (EXtensible Access Control Markup… | by Pamoda Wimalasiri | Identity Beyond Borders | Medium*, n.d.; *Attribute Based Access Control | CSRC*, n.d.; *EXtensible Access Control Markup Language (XACML) Version 3.0*, n.d.-a; *EXtensible Access Control Markup Language (XACML) Version 3.0*, n.d.-b; "Open Policy Agent," 2022; "Pure-Xacml," n.d.; "Understanding XACML Combining Algorithms," 2014; *Using Attribute-Based Access Control to Solve Role Explosion (Part 2) | Thoughtworks*, n.d.; *XACML - Wikipedia*, n.d.; Hu et al., 2015; Osborn & Servos, 2017b, 2017a; Sharma & Joshi, 2016; WANG et al., n.d.)

A beginner's guide to XACML. XACML (eXtensible Access Control Markup… | by Pamoda Wimalasiri | Identity Beyond Borders | Medium. (n.d.). Retrieved December 1, 2022, from https://medium.com/identity-beyond-borders/a-beginners-guide-to-xacml-6dc75b547d55

Attribute Based Access Control | CSRC. (n.d.). Retrieved December 11, 2022, from https://csrc.nist.gov/projects/attribute-based-access-control

EXtensible Access Control Markup Language (XACML) Version 3.0. (n.d.-a). Retrieved December 11, 2022, from https://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html

eXtensible Access Control Markup Language (XACML) Version 3.0. (n.d.-b). Retrieved December 11, 2022, from http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html#_Toc325047219

Hu, V. C., Kuhn, D. R., & Ferraiolo, D. F. (2015). Attribute-based access control. *Computer*, *48*(2), 85–88. https://doi.org/10.1109/MC.2015.33

Open Policy Agent. (2022). *GitHub*. https://github.com/open-policy-agent/opa

Osborn, S. L., & Servos, D. (2017a). *Current Research and Open Problems in Attribute-Based Access Control Article in ACM Computing Surveys ·*. https://doi.org/10.1145/3007204

Osborn, S. L., & Servos, D. (2017b). *Current Research and Open Problems in Attribute-Based Access Control Article in ACM Computing Surveys ·*. https://doi.org/10.1145/3007204

pure-xacml. (n.d.). *Www.Axiomatics.Com*. Retrieved December 11, 2022, from http://www.axiomatics.com/pure-xacml.html

Sharma, N. K., & Joshi, A. (2016). Representing Attribute Based Access Control Policies in OWL. *Proceedings - 2016 IEEE 10th International Conference on Semantic Computing, ICSC 2016*, 333–336. https://doi.org/10.1109/ICSC.2016.16

Understanding XACML combining algorithms. (2014). *Www.Axiomatics.Com*. https://www.axiomatics.com/blog/entry/understanding-xacml-combining-algorithms.html

Using attribute-based access control to solve role explosion (part 2) | Thoughtworks. (n.d.). Retrieved December 11, 2022, from https://www.thoughtworks.com/insights/blog/microservices/using-abac-solve-role-explosion-pt2

WANG, X., Fu, H., SINICA, L. Z.-A. E., & 2010, undefined. (n.d.). Research progress on attribute-based access control. *Ejournal.Org.Cn*. Retrieved December 11, 2022, from https://www.ejournal.org.cn/EN/abstract/abstract2290.shtml

XACML - Wikipedia. (n.d.). Retrieved December 11, 2022, from https://en.wikipedia.org/wiki/XACML