Research Manual

Web based XACML Graphical User Interface Policy Editor

Lee Wolohan C00157339

Supervisor: Dr. Joseph Kehoe

Version: 1.0

Date: 17th April 2015

Table of Contents

Document Description

The purpose of this document is to document the necessary research conducted to design and develop the web based XACML Graphical User Interface Editor. This document will cover the technologies used, including some of the libraries that were used throughout the project.

Technologies

The Java Standard Edition application programming interface (API) provides the core functionality of the Java language. It defines everything from the basic types and objects of the language to high-level classes that are used for networking, database access, security, graphical user interface development and XML parsing.

The Java Standard Edition also contains a virtual machine, development/deployment tools and many other libraries used in Java applications.

The Java Enterprise Edition platform is built on top of the Java Standard Edition platform. The Java Enterprise Edition platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, scalable, multi-tiered, reliable, and secure network applications. The enterprise needs are usually large-scale, transactional and designed to support critical business requirements.[1]

The Java Enterprise Edition platform is often referred to as J2EE (Java 2 Platform, Enterprise Edition).

ZK Framework

Web applications have evolved from static html pages, to dynamic html pages, to pages that use flash and finally to pages that use Asynchronous JavaScript and XML(Ajax) technology.

Ajax delivers the same level of interactivity and responsiveness as desktop applications. Ajax is based on the standard browser and JavaScript.

The ZK Framework is one of the leading enterprise Ajax frameworks, written in Java it is one of the best ways to build Java Web applications. It was developed by Potix.[2] The ZK Framework is an event driven, component-based framework to enable user interfaces for web applications. It includes an Ajax-based event-driven engine, an XML Interface language (XUL) and html components (XHTML).

In the ZK language Ajax is a behind the scenes technology. Low-level HTTP communication between the browser and server is handled by the framework.

Another main feature of the ZK Framework is the use of the XML User Interface Language (XUL) as a description language of graphical forms (ZUL). XUL lets you define forms as XML documents where individual tags correspond to controls on a form, making the process of designing web interfaces easier on the programmer.

Throughout my research of the ZK Framework I found a very useful resource in the development of ZK applications called the ZK Sandbox[3]. The ZK Sandbox provides many examples of many different layouts, input fields, activities and menus.

XML

Extensible Mark-up Language(XML) is a language that defines a set of rules for encoding documents in a format that is both human and machine readable. The XML language has no predefined tags. XML allows the user to define their own tags and their own document structure. The aim of XML is to implement simplicity, generality and usability across the Internet. It is used a lot in the representation of data structures in web services. XML is also the basic language for communication protocols.

XML characters are divided into mark-up and content. Strings that begin with < and end with > are considered mark-up. Strings that are not mark-up are classified as content.

There is three types of tags in XML:

1. <tag> : start tag.
2. </tag> : end tag.
3. <tag/> : empty element tags

XACML

eXtensible Access Control Mark-up Language(XACML) this is an access control policy language implemented in XML and a processing model to evaluate access requests according to the rules defined in policies.

XACML is an Attribute Based Access Control system. Attributes associated with a user, action or resource are inputs which decide whether a user may access a resource.
The response always includes one of four values:

1. Permit
2. Deny
3. Indeterminate(Error/Missing value)
4. Not Applicable(Can’t be answered by the service).

XACML version 2.0 was approved on 1st February 2005, followed by version 3.0 on the 10th August 2010.[4]

Policy Enforcement Point (PEP)

The file-system or web server doing the protecting.

The PEP forms a request based on the requesters attributes and the action they are trying to perform.

Policy Decision Point (PDP)

Looks at the policy that applies and decides whether to grant access or not. The answer is returned to the PEP, the PEP then allows or denies access to the requester. The PEP and PDP might be in the same system or they could be spread across several servers.[5]

Policy Elements

Policies and Requests both use Subjects, Resources, Environments and Actions:

- A Subject is the entity requesting access. A Subject can have one or more Attributes.
- The Resource is a data or system component. A Resource can have one or more Attributes.
- An Action specifies the type of access requested on the Resource. Actions can have one or more Attributes.
- An Environment can provide additional information.

Targets:

A target is a set of conditions for the subject, resource and action that must be met for the policy or rule to apply to a request. The rules are then evaluated to determine the decision and response.

Conditions:

Conditions exist in rules. Conditions are the same as a target except they can use a more advanced functions.

Obligations:

Obligations decide what must be applied before or after access is approved. If the Policy Enforcement Point cannot apply these obligations, the approved access must not be allowed.

Tags

The required tags for creating Policy, Request and Response were determined through the use of conformance data I requested from Primeur. A link to this conformance data can be found.[6]

Document Object Model Parser

The Document Object Model (DOM) parser loads the XML content into a Tree structure.[7] The DOM parser then iterates through the nodes and node lists to retrieve the content of the XML. The DOM parser provides application programming interfaces that allow to programmer to create, modify, delete and rearrange nodes.

There is four steps required to open and parse a document using the DOM parser in Java:

1. Create a document builder using the built in Document Builder Factory and parse the XML file to create a Document Object Model object.
2. Retrieve a list of elements from the Document Object Model using the built in NodeList, Node and Elements keywords.
3. Retrieve the attributes for each element and add them to the list associated with that element.
4. Iterate through the list of elements to verify the XML file was parsed correctly.

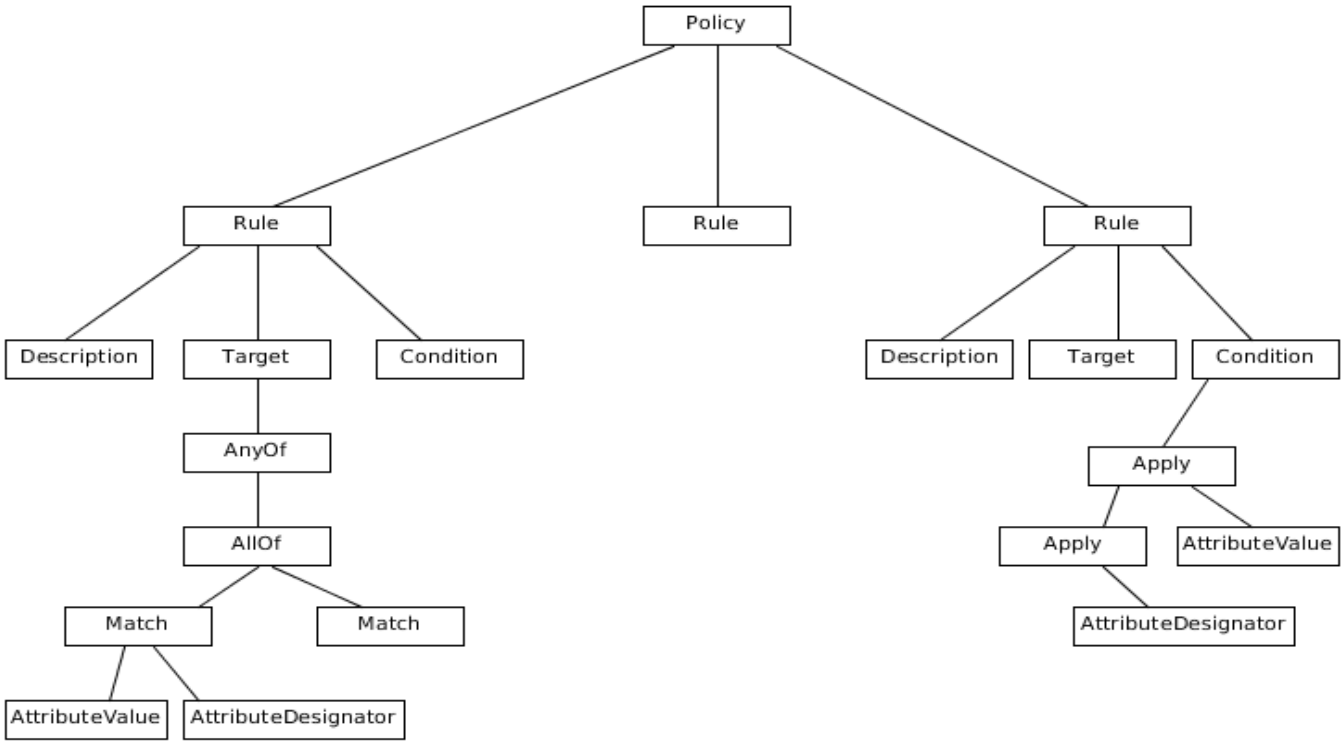
There is five steps required to parse and save a document using the DOM parser in Java:

1. Create a document builder using the built in Document Builder Factory.
2. Create a list of elements using the built in NodeList, Node and Element keywords.
3. Create the attributes for each element and add them to the list associated with that element.
4. Iterate through the list of elements to verify the Tree has been created correctly.
5. Add the Tree to the Document Builder Factory and write to the XML file.

Some of the node operations used in this project will be:

- Creating nodes
- Traversing nodes
- Searching for nodes
- Obtaining node content
- Creating attributes
- Inserting nodes

Dom Parser Structure



Apache Tomcat Servlet Container

Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies. Apache Tomcat creates a Java Http Web Server environment for the Java code to run in. It also provides a http server log which allows to user to see the connections necessary in a http web server. Apache Tomcat also provides server error messages to the developer which is very advantageous when attempting to identify the reasons for failure.

CodeMirror (Library)

CodeMirror is a text editor designed in JavaScript for web browsers. It is designed specifically for editing code, as a result it supports a number of different languages. CodeMirror is an open source project. It is used in development tools for Mozilla Firefox, Google Chrome and many other projects. [8] The browsers that support CodeMirror are listed below:

Firefox Version 4 and up

Chrome Any Version

Safari Version 5.2 and up

Internet Explorer Version 8 and up

Opera Version 9 and up

For this project I plan to use the XML component due to the fact that XACML files are stored in XML format.

Github Commands

clone initial repository: git clone [http address]

copy files into folder

git add .

commit: git commit -m "message to add"

push files to github: git push origin master

pull files from github: git pull origin master

References

[1]. Differences between Java EE and Java SE

http://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html

[2]. The ZK Framework by Andrzej Sekula
http://collaboration.cmc.ec.gc.ca/science/rpn/biblio/ddj/Website/articles/DDJ/2008/0802/080101as01/080101as01.html

[3]. The ZK Sandbox

http://www.zkoss.org/zksandbox/

[4]. Oasis eXtensible Access Control Mark-up Language by Bill Parducci, Hal Lockhart, Rich Levinson at Oracle.

https://www.oasis-open.org/committees/xacml/

[5]. Oasis A Brief Introduction to XACML

https://www.oasis-open.org/committees/download.php/2713/Brief_Introduction_to_XACML.html

[6]. Conformance Data Request from Primeur

https://drive.google.com/folderview?id=0B5v4tG69qG-bfmpzaUhtcXhZVU5UVUsxX1ZlalNBb3pDSEZycjZQTml2cUdYUDBCVEZHMGM&usp=sharing

[7]. The Document Object Model parser by mkyong

http://www.mkyong.com/java/how-to-read-xml-file-in-java-dom-parser/

[8]. CodeMirror documentation

https://codemirror.net/mode/index.html