Future Plans & Adoption

As all of you have known, I developed several projects on googlecode, one of the most important project is enterprise-java-xacml, which has been downloaded almost 2,000 copies, and been proved as the fastest XACML evaluation engine among all open source XACML implementations[i]. Another important project called chopsticks, which is a framework that combine all security services and make them work together. Following are some of my ideas about those projects, including future plans, possible adoption or donations.

Ideas for Existing Projects

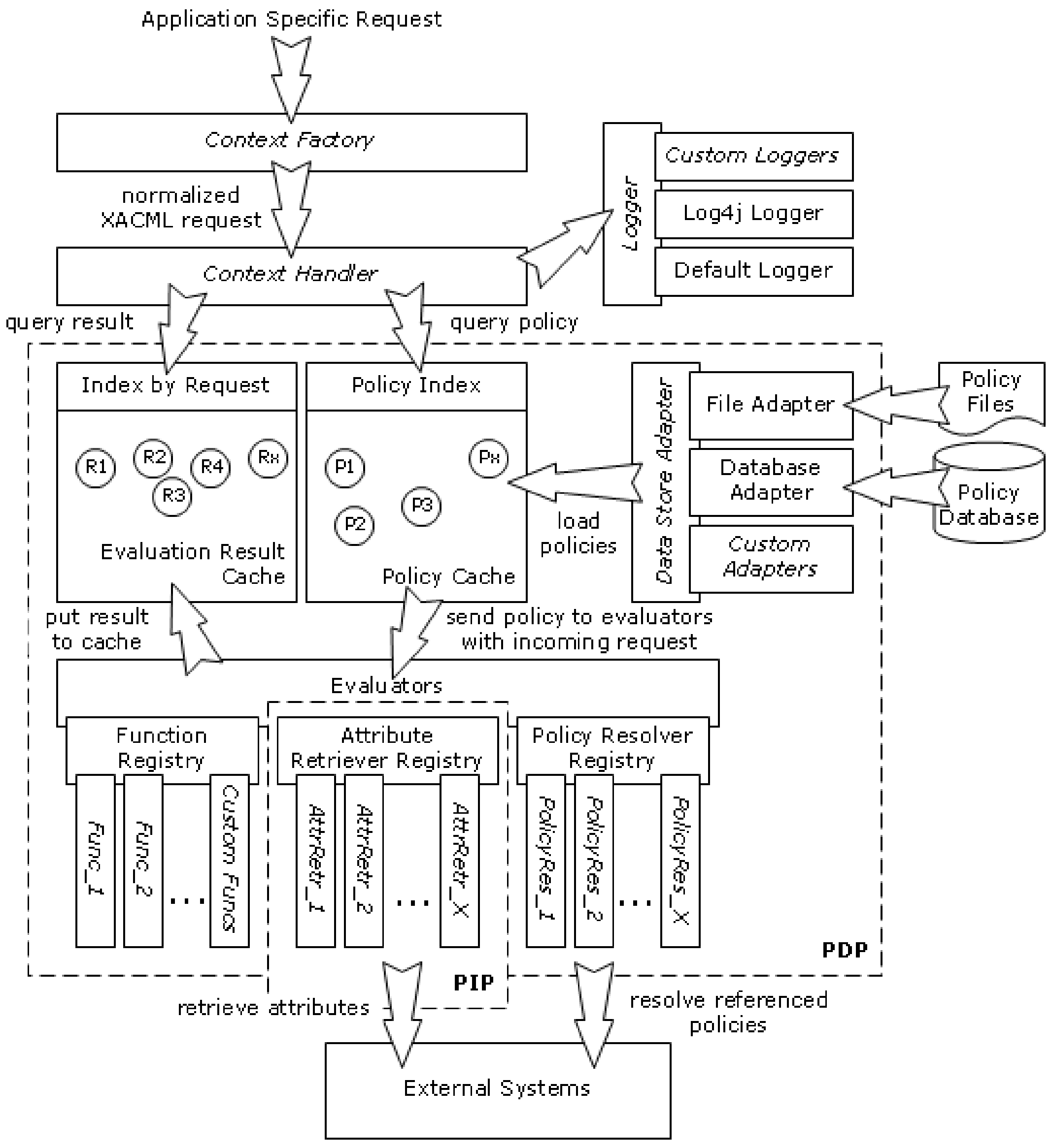
- **enterprise-java-xacml**

The first project I developed on googlecode, which is an implementation of XACML 2.0. In the early stage of 2005, I was working on some SOA projects, some of them need to integrate different applications together, the most important thing we need to do is to integrate business processes among applications. After the integrations are done, end-users of the applications only see a single application, so does the application administrator. But soon we found another issue, which is the security problem. Since all applications have their own authentication and access management, once the security policies changed (i.e. an employee has quit), we need make modifications (maybe code or configuration) to each related applications. We also can't enforce a single access policy among multiple applications. There isn't an existing access management solution for such a complex environment.

Fortunately, XACML moves forward into eyes. I realized the XACML is just what I want, so I searched all implementations on internet, and then I found Sun's implementation. Sun's implementation seems to be the most popular one, however it has many shortcomings. First, it doesn't have any cache mechanism for the policy or the evaluation result, this lead to a low performance for the policy evaluation; the second, it doesn't have an effective policy search mechanism, if an XACML request is coming, it just try each policy against the request, if there are thousands of policies in the system, it dramatically drops down the performance; the third, it doesn't define the situation that multiple policies match a single request, this situation isn't defined in XACML standard either, but it truly exists in real-life environment; the fourth, it read policies from local file, if one want save the policies to a database, he has to modify or enhance the implementation; the last, Sun's implementation lacks for providing extension mechanisms, such as attribute retriever, referenced policy resolver, and so on, such mechanisms will be very useful in a complex real-life environment.

So I wrote my own XACML implementation. My design point is to provide an extensible, scalable, faster, easy to use, easy to be integrated with applications architecture, and then it can be used in a complex enterprise environment. Since developers always integrate XACML engines to applications, and large applications always employs 3rd party products or libraries, to avoid conflicts between different versions of 3rd party libraries, I introduced minimum 3rd party libraries. Actually, I only introduced Log4j, and it can be replaced with my own logger if there is need.

Following diagram shows my implementation's architecture. Notes all components with ***italic*** can be customized (by provide custom implementation) by users. Apparently it's not perfect, but I want it to be a good start.



Enterprise Java XACML Implementation Architecture

- **xacml-poliy-editor**

A powerful XACML engine should have a powerful configuration tool. Existing open source XACML policy editors such as the UME-XACML-Editor requires users know well to the XACML standard, but in general, enterprise application users don't care about the standard, they care about the business. Also, the UME-XACML-Editor is a Java Swing application and it hard to be integrated with enterprise applications. Overall it's not better than a good XML editor such as Oxygen XML Editor.

After discussed the issue with Hal Lockhart, the co-chairs of XACML TC, I decided to develop a set of PAP (Policy Administration Point) APIs and a pluggable, business oriented XACML policy editor. Hal gives me many good suggestions. He also pasted all his ideas to the XACML mail list.

For now, the project is in planning stage. I'm now working on XACML engine's re-factory, which is to export PAP APIs definitions.

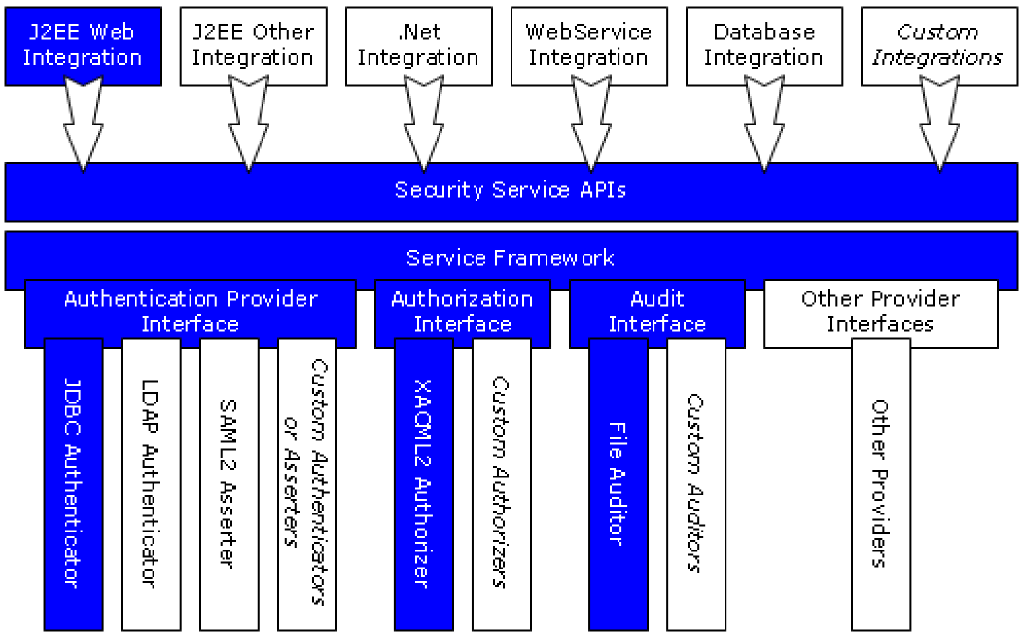
- **chopsticks**

Chopsticks is a framework that to combine multiple security services such as authentication, authorization and auditing services and make them work together. Chopsticks provides a set of security service APIs, application developers can make calls to them and then can enforce the features from the underlying security services to the target applications.

Chopsticks employs service provider design pattern, application developers may make use of existing providers, also may develop their own providers and then plug them into chopsticks. I have developed several providers: SQL authenticator, file based auditor, and XACML provider. One of the most important tasks in the future for this project is to develop more and more providers, such as SAML2.0 provider.

Beside the core architecture, chopsticks also focus on provide application integrations. In general, if an application need to employ a 3rd party mechanism such as my XACML engine, the application developers will responsible for modify the application and make calls to it. If such mechanism changed in the future, the application developers have to make changes to the application again to maintain the consistency. Chopsticks aims to resolve the problem, it provides standard ways to integrate with applications. By this way, applications need no modifications, application developers also don't care about the integration. Chopsticks provides a set of OOTB (Out Of The Box) application integrations, such as the Tomcat integration, WebLogic Server integration and so on. For now, only the Tomcat integration is developed, all other integrations will be developed in the future.

Following diagram shows chopsticks' architecture. Notes, all blocks filled with white color have not developed yet for now.



Chopsticks Architecture

- **other projects**

All my other projects on googlecode will serve chopsticks as service provider, such as xacml3 or saml2. For now, they are in planning stage.

Future Plans

The project chopsticks and enterprise-java-xacml have provided basic security features to applications. Beside the "urlsample" that I have provided for chopsticks in SVN repository, I have written another more really sample called "petstore" to illustrate all the features that chopsticks provided. This sample is not in current SVN repository, I will check it in soon. In this sample, you can see with only a few configuration steps (no any code changes), a normal web application can be protected with authentication, authorization services provided by chopsticks. Also, all accesses to the web application are audited.

The "petstore" sample shows the possible futures for the project - not only to protect web applications, but also to protect all other applications. And we can also provide many other security services which they may come from 3rd party organizations. The most important advantage we get is the ability to manage multiple applications' security in a distributed environment. Current projects only provided the basic architecture for security services, a distributed architecture will be developed in the future. For the project users, my idea is to provide OOTB services for them, I want they use my product with minimized custom development or modifications.

Recently, a new software type called "Entitlement Management" comes into eyes. What I want is exactly to develop a scalable, extensible and powerful Entitlement Management solution.

So to achieve all above objectives, at least following things need to be done in next several months,

1. A set of PAP APIs should be developed.
2. An XACML policy editor should be developed.
3. An administration program should be developed to create and distribute configurations/policies for multiple applications in distributed environment.
4. More integrations should be developed, such as WebLogic Server integration, WebSphere Application Server integration and so on.
5. A web service layer will be created to expose the security service APIs, then the applications written by C/C++/.Net or other languages can make use of features that chopsticks provided.
6. Documents for all the projects should be provided.
7. More samples should be developed to guide users how to use the projects.

Adoption or Donations

Since some tasks in the plan require developers work and communicate closely, I want hire some people in my city, and then open an office for us, then we can work together closely. So any donation and adoption, or venture capital is welcome! If you are interest to help me on my project, please send email to ppzian@gmail.com, I will send you my detailed development plan and discuss it with you.

[i] Fatih Turkmen and Bruno Crispo "Performance Evaluation of XACML PDP Implementations", 2008