White Paper:

Location, Location, Location

Where You Put Your Identity Attributes Matters

Role-based versus attribute-based access control

Around 40 years ago computer systems started to deliver multiple applications to multiple users and as a result data security emerged as an issue. To address this, software developers and system administrators began to develop different types of access control to ensure that only authorized users could access data and resources. One such method of access control is role-based access control (RBAC).

In RBAC administrators create roles, typically linked to job functions within their organization, and build access control policies using those roles. Users are then assigned one or more roles based on their job within the organization.

RBAC makes access management considerably easier in environments with large numbers of users and has been rather successful. However, it can become unmanageable over time. A common problem is “role explosion”, which occurs when the desire for an ever-finer level of granularity in access control results in roles being defined to cater for specific operations, data elements, or obscure access scenarios.

Perhaps as a result of these limitations attribute-based access control (ABAC) is now gaining traction. In the ABAC model access control policies are based on the properties of the user and resource. Whereas RBAC introduces a layer of abstraction which can become a bottleneck, ABAC uses information that is inherent to the users and resources being evaluated in order to make access control decisions.

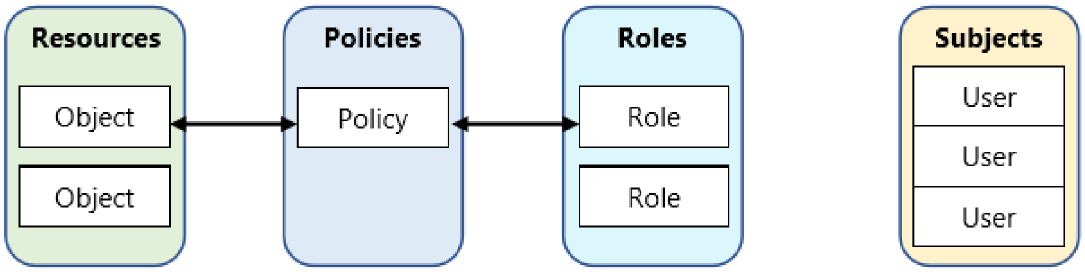
ABAC is the most suitable model for modern environments because it can operate as long as the necessary information about the resource and user is known when the access decision is made. There is no need to assign users to roles in advance.

Attribute-based access control and XACML

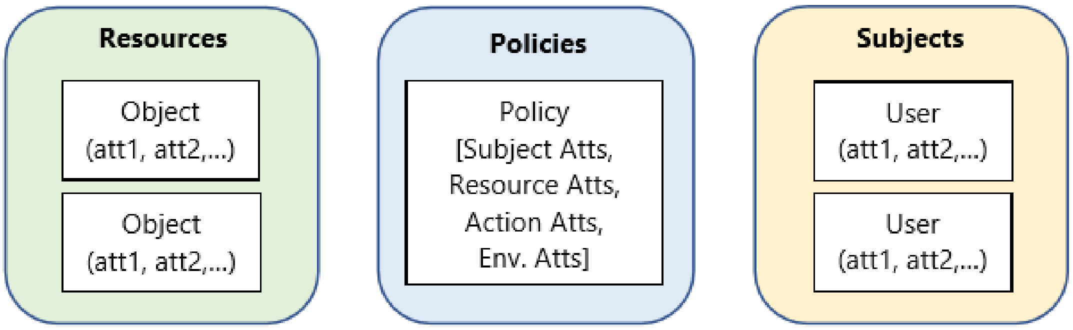
eXtensible Access Control Markup Language (XACML) is an OASIS standard for expressing authorization policies and processing authorization requests. XACML is fundamentally an ABAC system, although it can also be used to implement RBAC.

XACML Policies

In other access control models' policies are usually associated with specific resources, users or roles. For example, a typical RBAC policy (illustrated below) associates resources directly with roles. When such a policy is evaluated the user's role must be known.



XACML ABAC policies (illustrated below) are different because they are not tied to specific users, roles or resources. Instead, XACML policies identify users and resources through their *attributes*.

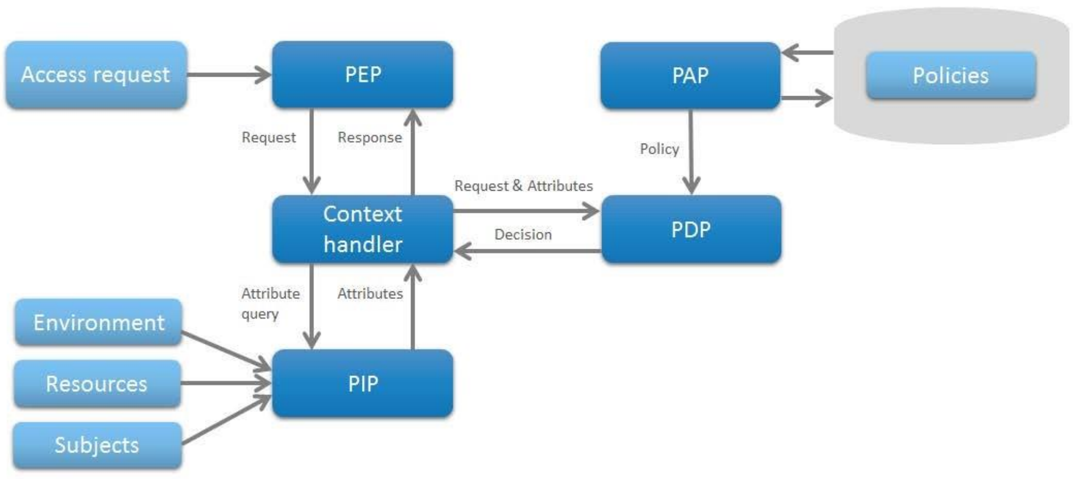


XACML policies usually contain attributes associated with the resource (the thing being accessed), subject (the user attempting to access the resource), para-action (what the subject wants to do with the resource) and environment (when, where and how access is being attempted), although these attribute categories are extensible. When an XACML policy is being evaluated, information about the resource and subject must be available at the time the access decision is made.

XACML processing

The XACML authorization system is made up of the policy enforcement point (PEP), context handler, policy decision point (PDP), policy administration point (PAP) and the policy information point (PIP). It is the role of the PDP to determine if an authorization request (provided by the PEP) should be permitted. The PDP leverages XACML policies (provided by the PAP) to make this decision. Since the XACML policies are attribute-based, the PDP requires resource, subject, action and environment attributes in order to make a decision. It is the role of the context handler to provide these attributes to the PDP, which it will source from information supplied by the PEP and by requesting attributes from the PIP.

The following diagram provides a simplified view of the XACML data flow model.



The XACML data flow model is non-normative. XACML does not prescribe how attribute information should be made available to the context handler. It is not unusual for this information to be collected from data repositories.

XACML's dirty little secret

The attribute-based nature of XACML is what makes it extremely flexible and usable in many situations. However, it also implies a reliance on an "economy of attributes".

If an authorization request does not contain all of the attributes an authorization server needs in order to evaluate its policies, then the authorization server must obtain those attributes from elsewhere.

The idea of retrieving attributes from existing enterprise repositories is attractive. It reduces data duplication and therefore the cost of managing information in multiple locations.

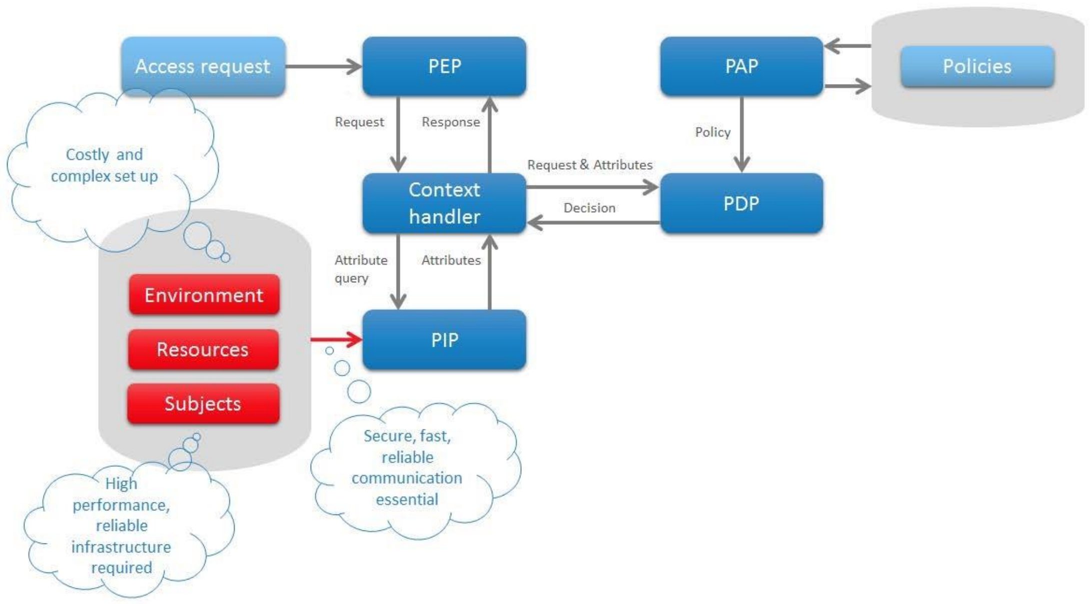
However, the attributes used to evaluate access control requests are both hugely important and potentially vulnerable in this type of authorization solution. Therefore, when considering how an authorization server accesses attributes, the following must be considered:

- The attribute sources must contain reliable data with known provenance.
- The attributes sources must contain high quality information that can reliably be used for security decisions.
- There must be well defined attribute lifecycles and administrative workflows.
- Communication with attribute sources must be secure.
- Attribute sources must be suitably authenticated to ensure information is being obtained from a trusted source.
- The attribute sources and the authorization model they use to protect themselves must be secure. Attribute sources are a significant point of failure within any ABAC system.
- The attribute sources must be high performance and able to tolerate the additional loads required to facilitate the authorization service.
- The authorization service must have persistent attribute stores.
- The attribute strategy must allow for new attributes to be defined, linked to existing users and resources, and made available to the authorization service.

As a result of all this, requiring your authorization system to leverage an external repository that maintains attributes for a purpose other than to enforce authorization decisions is not a good idea. The problems get even worse when the authorization service obtains attributes from multiple repositories. For this to be a viable solution, each data source must be trusted, secure and maintain high quality information, while being able to provide attributes at high speed and in a secure and reliable manner.

All About Virtual Directories and Centralized Metadirectories

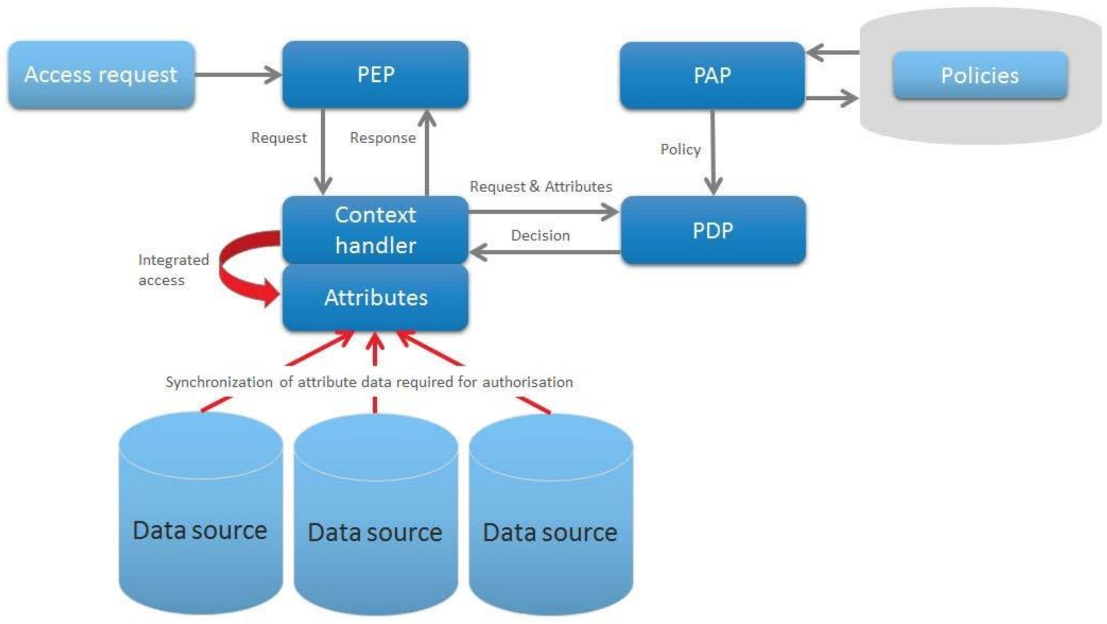
Rather than obtaining attributes from multiple repositories, many organizations use new or existing centralized virtual directories or meta-directories to provide information to the authorization server from a single location.



This type of single source solution makes the authorization service easier to configure and may alleviate some of the availability and performance issues associated with a multi-source environment. However, although the authorization service now only needs to obtain information from a single source, the overall system is more complex and more expensive. The problem has simply been shifted onto another component without necessarily addressing any of the larger security concerns.

Putting attributes in their place

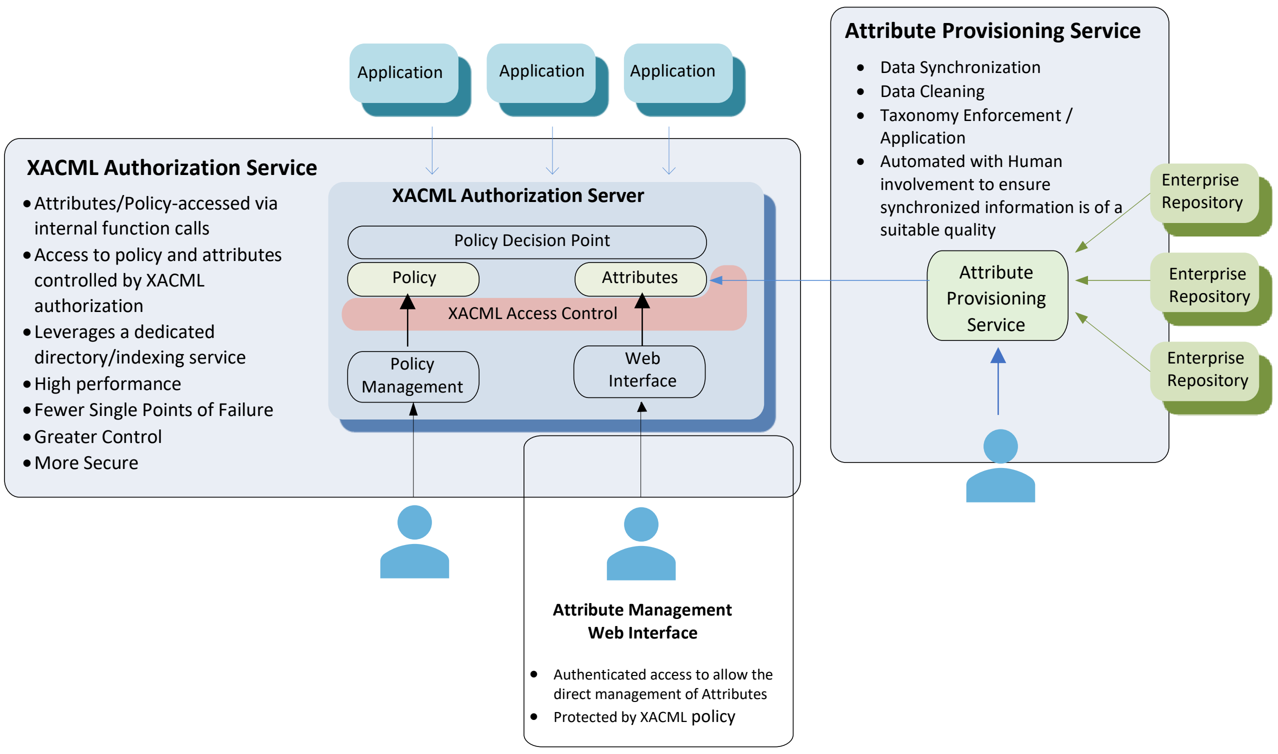
Clearly, retrieving attributes from an external source slows the system down, causing application performance to suffer. So, to provide the necessary security and performance, an ABAC authorization service should include integrated storage of the attributes and policies it needs to make authorization decisions.



An authorization solution with integrated attribute storage offers many benefits:

- A secure, dedicated, high-performance and highly available store for authorization attributes.
- The ability to maintain a high level of attribute quality and consistency without affecting other enterprise systems.
- The ability to tightly control who has access to authorization attributes.
- The ability to maintain additional attributes required for authorization.

Of course, if attributes are going to be maintained within the authorization solution, then proper facilities should be put in place to allow them to be managed. A good data synchronization solution will be able to observe the data being synchronized and normalize the information, as well as initiate approval workflow processes for any information that should not be added automatically to the authorization solution.


Application
Application
Application
XACML Authorization Service
Attributes/Policy-accessed via internal function calls
Access to policy and attributes controlled by XACML authorization
Leverages a dedicated directory/indexing service
High performance
Fewer Single Points of Failure
Greater Control
More Secure
XACML Authorization Server
Policy Decision Point
Policy
Attributes
XACML Access Control
Policy Management
Web Interface
Attribute Provisioning Service
Data Synchronization
Data Cleaning
Taxonomy Enforcement / Application
Automated with Human involvement to ensure synchronized information is of a suitable quality
Attribute Provisioning Service
Enterprise Repository
Enterprise Repository
Enterprise Repository
Attribute Management Web Interface
Authenticated access to allow the direct management of Attributes
Protected by XACML policy

ViewDS Access Sentinel

Access Sentinel is an authorization service with an integrated attribute store, that allows applications to externalize their authorization decisions.

It is built on ViewDS Directory Server, which allows both identity and policy information to be stored in a single, replicated repository. This unique capability provides real advantages in terms of efficiency, security and scalability.

Access Sentinel eliminates the network latency and poor security inherent in more traditional externalized authorization services.

In summary, Access Sentinel:

- Provides an ABAC and RBAC XACML 3.0 authorization service
- Contains an integrated attribute store, which is secure, dedicated, high performance and highly available
- Overcomes the security and performance limitations of traditional authorization solutions
- Provides a high level of attribute quality and consistency without affecting other enterprise systems
- Strictly controls access to authorization attributes
- Is able to maintain the additional attributes required for authorization

ViewDS Identity Bridge

Identity Bridge is a powerful, highly configurable and easy to use data synchronization and provisioning solution. It provides:

- Attribute data synchronization
- Data cleaning
- Taxonomy application and enforcement