PoC Use Cases

1. Coarse-Grain Authorization :

Rational: Authorization is done based only on subject, resource and action

Web app scenario: Apply for Loan
Policy Name: Course-Grain-Usecase
Policy Rule: A loan can be applied only if the username is bob

Test Case:

Permit - bob
Deny - alice

- Bob is allowed to apply for a loan because his name is specified in the Course-Grain-Usecase policy
- Alice is not allowed to apply for a loan because she is not considered in the policy for the Loan resource.

2. Fine-Grain Authorization

Rational: Authorization is done not only on subject, resource and action, but also on user attributes like Role.

Web app scenario: Do a Payment
Policy Name: Fine-Grain-Usecase
Policy Rule: A payment can be done only if the user is in the Gold role.

Test Case:

Permit - bob
Deny - peter

- Bob is allowed to to do a Payment because he has the ‘Gold’ Role.
- Peter is not allowed to do a payment because he does not have the ‘Gold’ Role.

3. Multiple Roles based Authorization

Rational: Resource authorization requires more than one role.

Web app scenario: Credit Card Reversal
Policy Name: Multiple-Roles-Usecase
Policy Rule: To do credit card reversal both the CardCenterLead and CardCenterRepresentative should be assigned

Test Case:

Permit: alice
Deny: bob

- Alice is allowed to do credit card reversal because she has both CardCenterLead and CardCenterRepresentative roles.
- Bob is denied to do the reversal because he does not have the CardCenterLead role.

4. Hierarchical User Roles based Authorization

Rational: A user from a parent role has all the permissions of a child role.

Web app scenario: Create Account
Policy Name: ManagerRole, ManagerPermission, EmployeeRole, EmployeePermission, SuperviserPermission
Policy Rule: An account can be created only if the user is of Employee role or in a parent role.

Test Case:

Permit: mike
Deny: peter

- Mike (Manager) can create an account because Esther (Employee) can do the same.
- Peter cannot create an account because he does not have the Manager role.

5. Hierarchical Resources based Authorization

Rational: A subject can do an action on a resource only if he can act on another resource.

Web app scenario: Close Account
Policy Name: ManagerRole, ManagerPermission, EmployeeRole, EmployeePermission, SuperviserPermission
Policy Rule: An account can be closed only if the user has permission to close a debit-card

Test Case:

Permit: mike
Deny: esther

- Mike (Manager) can close and account because the debit card closure is allowed for Supervisors and Mike has that permission.
- Esther cannot close an account because she does not have the Supervisor permissions, thus cannot close a debit card.

6. Conflicting Entitlements

Rational: When two policies or rules provide conflicting results (i.e. permit and deny) then a precedence should be defined using a result combining algorithm

Web app scenario: Money Transfer
Policy Name: Overlapping - Entitlements-Usecase
Policy Rule: Money transfer is allowed for role 'Gold' and denied for role 'Silver'

Test Case:

User: *marker*

- Marker has both Gold and Silver roles.
- If the rule combining algorithm is *urn:oasis:names:tc:xacml:3.0:rule-combining-algorithm:permit-overrides* result is permit.
- If the rule combining algorithm is *urn:oasis:names:tc:xacml:3.0:rule-combining-algorithm:deny-overrides* result is deny.

7. External Data

Rational: User data should be fetched from external sources.

Web app scenario: Request Cheque Book
Policy Name: External-Data-Usecase
Policy Rule: Cheque Book request is allowed only if the available account balance is greater than 50000.

Test Case:

Permit: peter
Deny: bob

- Peter is permitted to request a cheque book because his account balance is 70000
- Bob is not permitted to request a cheque book because his account balance is 30000

8. Reverse Lookup

Web app scenario: Reverse Lookup

Test Cases:

1. User Based:

   Tenant: Tenant Domain <leave blank for Super Tenant>
   Subject Type: User
   User / Role Name*: mike (or any username)
   Subject Id*: urn:oasis:names:tc:xacml:1.0:subject:subject-id

1. Role Based:

   Tenant: Tenant Domain <leave blank for Super Tenant>
   Subject Type: Role
   User / Role Name*: Manager (or any role)
   Subject Id*: http://wso2.org/claims/role

NOTE: to test the tenant (i.e. application group) reverse look up, a tenant should be created in the Identity Server and some policies should exist in it. According to the current implementation of the web app tenant admin's username should be "admin" and password should be "123456".