Blockchain based Access Control Framework
Outline

• Background
  – Access Control
  – XACML

• Blockchain based Access Control
Background: Access Control and XACML
Access Control

Technique to decide whether a **Subject** requesting to perform an **Action** on a **Resource** in a given **Context** holds the right to perform it.
Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right to perform it.
Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it.
Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it.
Access Control

Technique to decide whether a **Subject** requesting to perform an **Action** on a **Resource** in a given **Context** holds the right to perform it.

Subject \[\rightarrow\] Performs Actions

Resources
Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right to perform it.
Access Control Models

- Mandatory
- Discretionary
- Identity based
- Role based
- Intent based
- Attribute based
- ........
Attribute Based Access Control (ABAC)

An access control method where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions.

Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publication 800-162
Attribute Based Access Control (ABAC)

- A subject is a human user or Non Person Entity, such as a device that issues access requests to perform operations on objects. Subjects are assigned one or more attributes.

- An object is a system resource for which access is managed by the ABAC system, such as devices, files, records, tables, processes, programs, networks, or domains containing or receiving information. It can be the resource or requested entity, as well as anything upon which an operation may be performed by a subject including data, applications, services, devices, and networks.

- An operation is the execution of a function at the request of a subject upon an object. Operations include read, write, edit, delete, copy, execute, and modify.

- Policy is the representation of rules or relationships that makes it possible to determine if a requested access should be allowed, given the values of the attributes of the subject, object, and environment conditions.
Attributes

- Attributes represent characteristics of the
  - Subjects
  - Resources
  - Environment
  - Actions

- Examples:
  - Subject
    - Company the subject works for
    - Role (e.g., in a company: Worker, Employee, Executive, CEO...)
    - Projects assigned to the subject
    - Clearance level
  - Resources
    - Owner/producer
    - Number of copies of a document
    - Project of a document
    - Security classification
Attributes

Other examples?
Attributes

• The access context is represented by Attributes

• **Immutable Attributes:**
  - Rarely change their values
  - Their update requires the intervention of an administrator
  - Examples:
    • Subject/Resource ID
    • Subject Role in a company
      - Changes because of a career advancement
    • Permanent storage capability of a Resource
      - Changes because of hardware updates
Mutable Attributes

• **Attributes are Mutable** when they **change their value** over time because of the **normal operation** of the system:
  - Explicit update commands included in the security policy (Attribute Updates)
  - Actions executed by the users which affect the access context
  - Actions executed by resource management systems
  - Environmental factors which change
  - .......
• The attribute update is managed by the system, no admin intervention is required
Examples of Mutable Attributes

- **Subject**
  - Number of accesses in progress
  - Physical position
  - E-wallet
  - Reputation
  - On/off duty

- **Resource**
  - Number of copies of a document
  - Number of accesses in progress on the resource
  - List of the users who are currently accessing the resource

- **Environment**
  - Date and Time
  - System state
Examples of Mutable Attribute Updates

- **Subject**
  - Number of accesses currently in progress
    - The policy includes a statement which increases the value when a new access is granted and a statement which decreases the value when an access terminates
  - Physical position
    - The user moves from one place to another
      - Geo-localization systems

- **Resource**
  - Number of copies of a document
    - The policy includes a statement which increases the value when a user is allowed to create a new copy

- **Environment**
  - Date and Time
    - Obviously!!
Mutable Attributes

Other examples?
Attribute Authorities/Managers

• Manage a set of attributes, are invoked:
  – To get the current value of the attributes
  – Sometimes also to request to update the current value of the attributes

• Identity Providers could act as Attribute Providers, for instance producing SAML assertion including
  – Authentication statement
  – Attribute Statement

• Existing services can be used as Attribute Managers
  – LDAP
  – Active Directory
  – DB
  – ..
Attribute Authorities/Managers

- Authorization System itself
  - Attribute values managed through Attribute Updates

- Resources/ Resource Management Systems
  - A document could embed some attributes as metadata
    - E.g., the document producer
  - The management system of the resource could provide information about the resource (workload, number of resources, number of accesses)
    - E.g., a Cloud Management System

- Other external systems
  - Attribute Authorities
  - LDAP servers
  - Active Directory
Attribute Representation: Security Assertion Markup Language 2.0 (SAML)

- OASIS standard for the exchange of data among domains concerning: authentication, authorization, and attributes

- Actors
  - Identity/attribute provider: produces SAML assertions
  - Service provider: consumes SAML assertions

- SAML Assertions embed:
  - Authentication statement: the subject has been authenticated with a given procedure at a given time
  - Authorization statement: the access request has been permitted or denied
  - Attribute statement: the subject holds this set of attributes
Esempio di SAML 2.0 Assertion

<saml:Assertion
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    ID="b07b804c-7c29-ea16-7300-4f3d6f7928ac"
    Version="2.0"
    IssueInstant="2004-12-05T09:22:05">
    <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
    <ds:Signature
    <saml:Subject>
        <saml:NameID
            Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
            3f7b3dcf-1674-4ecd-92c8-1544f346baf8
        </saml:NameID>
    </saml:Subject>
</saml:Assertion>
Esempio di SAML 2.0 Assertion

<saml:SubjectConfirmation
  Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
  <saml:SubjectConfirmationData
    InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
    Recipient="https://sp.example.com/SAML2/SSO/POST"
    NotOnOrAfter="2004-12-05T09:27:05"/>
</saml:SubjectConfirmation>
</saml:Subject>
<saml:Conditions
  NotBefore="2004-12-05T09:17:05"
  NotOnOrAfter="2004-12-05T09:27:05">
  <saml:AudienceRestriction>
    <saml:Audience>https://sp.example.com/SAML2</saml:Audience>
  </saml:AudienceRestriction>
</saml:Conditions>
Esempio di SAML 2.0 Assertion

<saml:AuthnStatement>
  AuthnInstant="2004-12-05T09:22:00"
  SessionIndex="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
Esempio di SAML 2.0 Assertion

<saml:AttributeStatement>
  <saml:Attribute
    xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
    x500:Encoding="LDAP"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
    Name="role">
    <saml:AttributeValue
      xsi:type="xs:string">member</saml:AttributeValue>
    <saml:AttributeValue
      xsi:type="xs:string">staff</saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
</saml:Assertion>
Access Control Policy

- Set of rules having an effect (permit/deny) defined in terms of conditions on attributes of subjects, resources, actions, and environment

+ Combination algorithms to decide the precedence among those rules

Policy example (Natural Language):

Documents can be read if the value of the attribute ROLE of the Subject is “employee” and if the value of the attribute PROJECT BELONGING TO of the Resource is equal to (one of) the value of the attribute PROJECT ASSIGNED TO of the Subject
Access Control Policy

ROLE = "employee"

PROJECT ASSIGNED TO = "ProjectA", "ProjectB"

PROJECT BELONGING TO = "ProjectA"
Access Control Policy
Access Control Policy

ROLE = “employee”

PROJECT ASSIGNED TO = “ProjectA”, “ProjectB”

PROJECT BELONGING TO = “ProjectC”
Access Control Policy
XACML

Extensible Access Control Markup Language
3.0

OASIS
Extensible Access Control Markup Language 3.0 (XACML)

XACML defines:

- A XML-based Language to express Attribute based Access Control Policies
- A reference architecture for the Access Control Framework

eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01.
OASIS Standard incorporating Approved Errata. 12 July 2017
Extensible Access Control Markup Language 3.0 (XACML): Policy Language
Extensible Access Control Markup Language 3.0 (XACML): Policy Example

<Policy PolicyId="PolicyCNR" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable" Version="3.0">
  <Description>PolicyForCNR</Description>

  <Rule Effect="Permit" RuleId="authorization_1">
    <Target>
      <AnyOf> <AllOf>
        <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">687fff544f346baf8</AttributeValue>
        </Match>
      </AllOf> </AnyOf>
    </Target>
  </Rule>
</Policy>
Extensible Access Control Markup Language 3.0 (XACML): Policy Example

<Condition>
</Condition>

</Rule>
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

1. Define policy

Service Requester

Policy Requester

Policy Enforcement Point

Obligations Service

Policy Administration Point

Attribute Manager Resources

Attribute Manager Subjects

Subject

Resource Owner

Environment

Attribute Manager Environment

Context Handler

Policy Information Point

Resource

Attribute

Manager

Owner

Manager

With the given text and diagram, the document seems to discuss the Extensible Access Control Markup Language (XACML) 3.0 and its reference architecture. The diagram illustrates the flow of access control decisions, starting with a request from a Service Requester, going through a Policy Enforcement Point, and ultimately resulting in the execution of obligations. The process involves interaction with various entities such as Subjects, Resources, and Attribute Managers. The text provides a structured overview of the components and their roles in the XACML architecture.
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- Policy Enforcement Point
  - Embedded in the resource to be protected
  - Intercepts access request of users
  - Ask the evaluation of the policy for that request
  - Receives an access response and enforces it by
    - PERMIT: executing the access request on the resource
    - DENY: not executing the access request
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- **Context Handler**
  - Front-end of the access control system
  - Receives an access request from the PEP
  - Returns an access response to the PEP
  - Coordinates the other components to implement the decision process
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- **Attribute Managers (AMs)**
  - Manage the attributes required for the policy evaluation
  - Each application scenario has its own AMs
  - Each AM has its own API and protocol
  - Local or Remote
  - E.g., LDAP server
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- Policy Information Points
  - Interfaces with AMs to retrieve the attribute values
  - Each PIP is customized for a specific AM
- API and protocol
- Attribute format
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- **Policy Decision Point**
  - Evaluates the policy against the access request and exploiting the attribute values collected by PIPs
Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture

- Policy Administration Point
  - Repository of available policies
  - Queried by the PDP to get the policy to be evaluated