HOCHSCHULE FURTWANGEN UNIVERSITY HFU

An Attribute Based Access Control Model for RESTful Services

Agenda

- Foundations
- eXtensible Access Control Markup Language (XACML)
- RestACL
- Test
- Conclusions

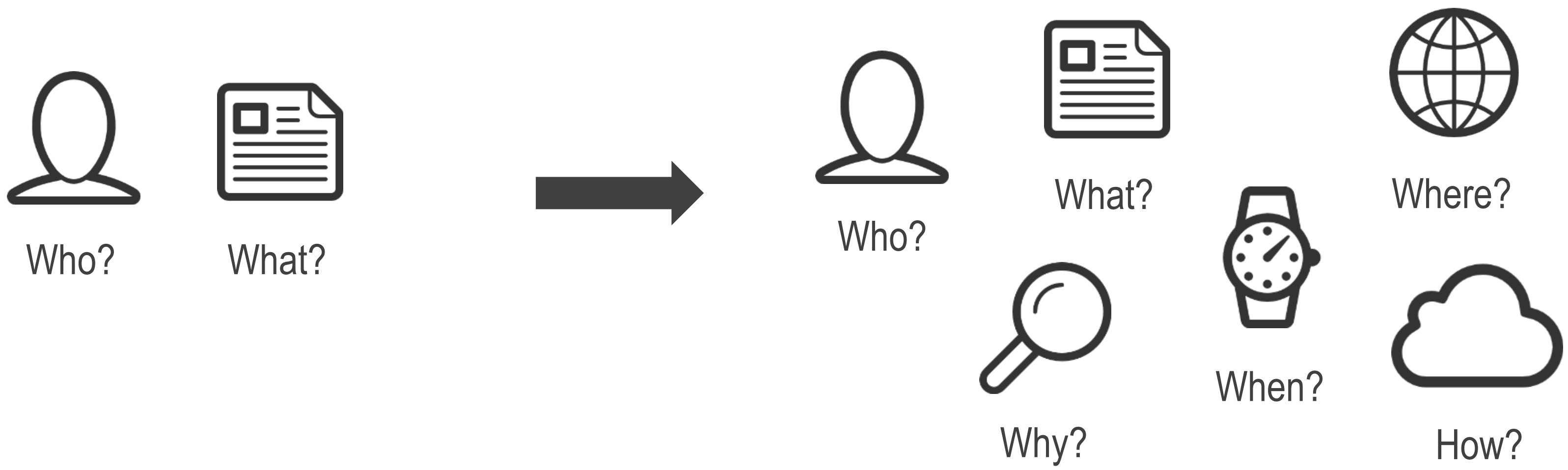
REST – Overview

- Architectural Style (Distributed Systems and Services)
  - T. R. Fielding. Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine, 2000
- Web Service based on HTTP
- 4 Core Concepts
  - **Resource Orientation**
  - Representations of Resources
  - Uniform Interface
  - Stateless Communication

ABAC – Motivation

Classic Access
Control Mechansim
(RBAC, ACL)

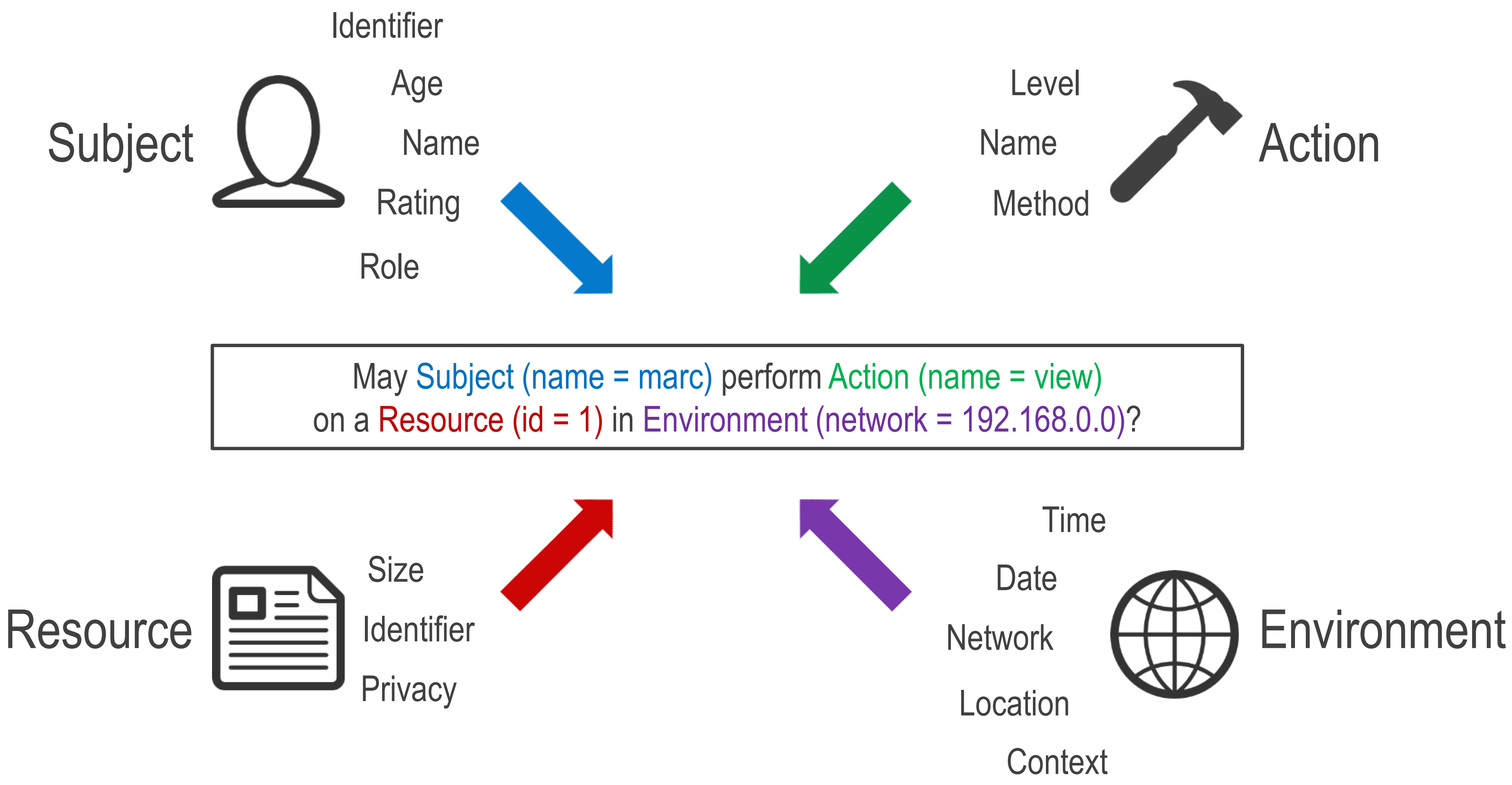
New Access
Control Mechansim
(ABAC)



“By 2020, 70% of all businesses will use ABAC as the dominant mechanism”

Source: Gartner, Identity and Access Management Predictions, Nov. 2013

ABAC – Idea

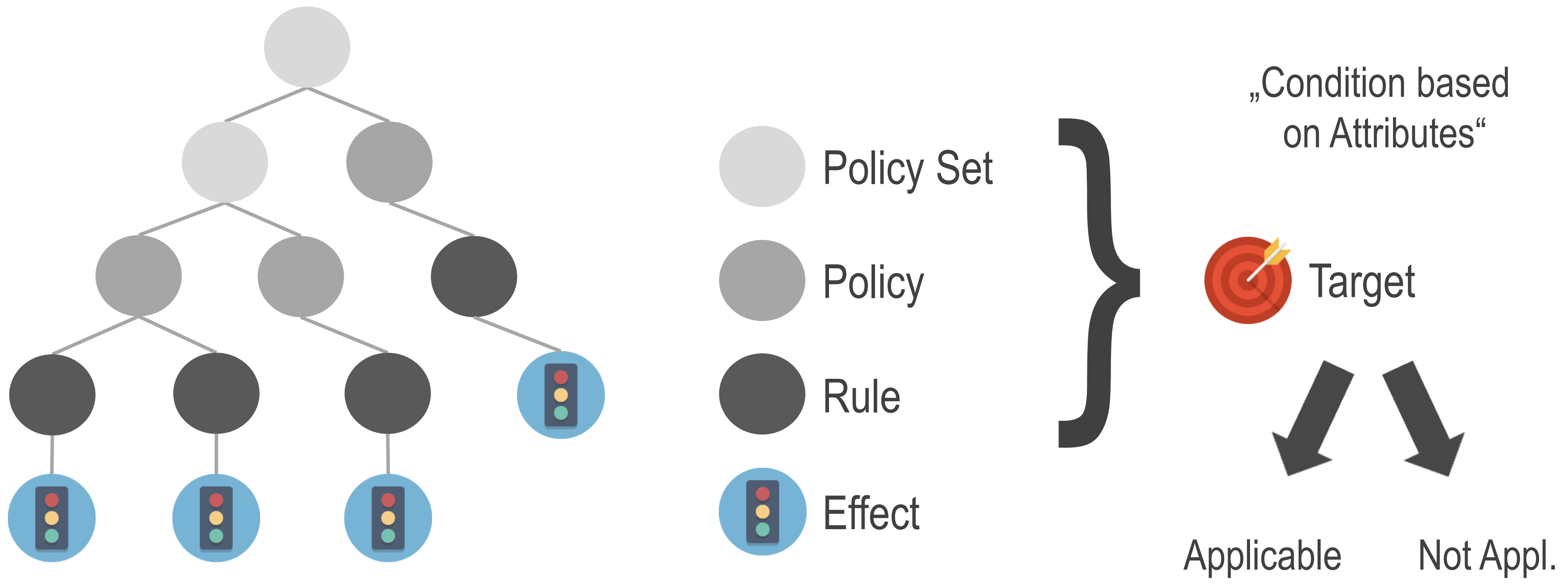


May Subject (name = marc) perform Action (name = view)
on a Resource (id = 1) in Environment (network = 192.168.0.0)?

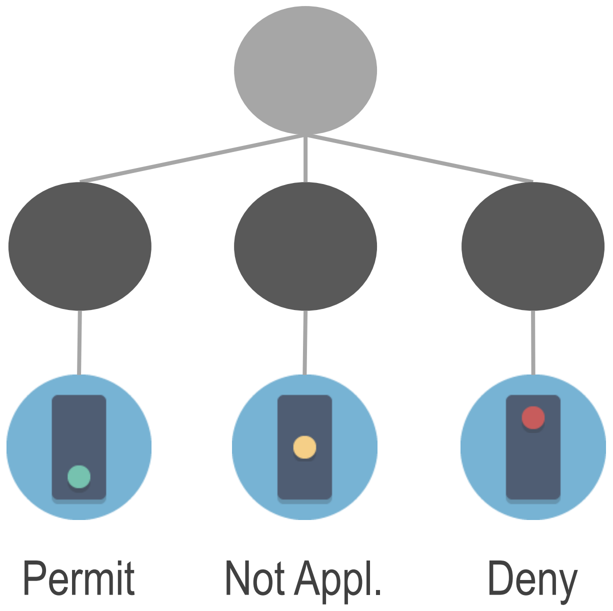
XACML – Overview

- OASIS Standard
  - http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html
- Latest Version: 3.0
  - Published 2013
- XACML defines
  - Architecture
  - Policy Language
  - Request/Response Language

XACML – Policy Language



XACML – Combining Algorithms



Combining Algorithms

- PermitOverrides
- DenyOverrides
- FirstApplicable
- OnlyOneApplicable
- …

XACML – Example (simplified)

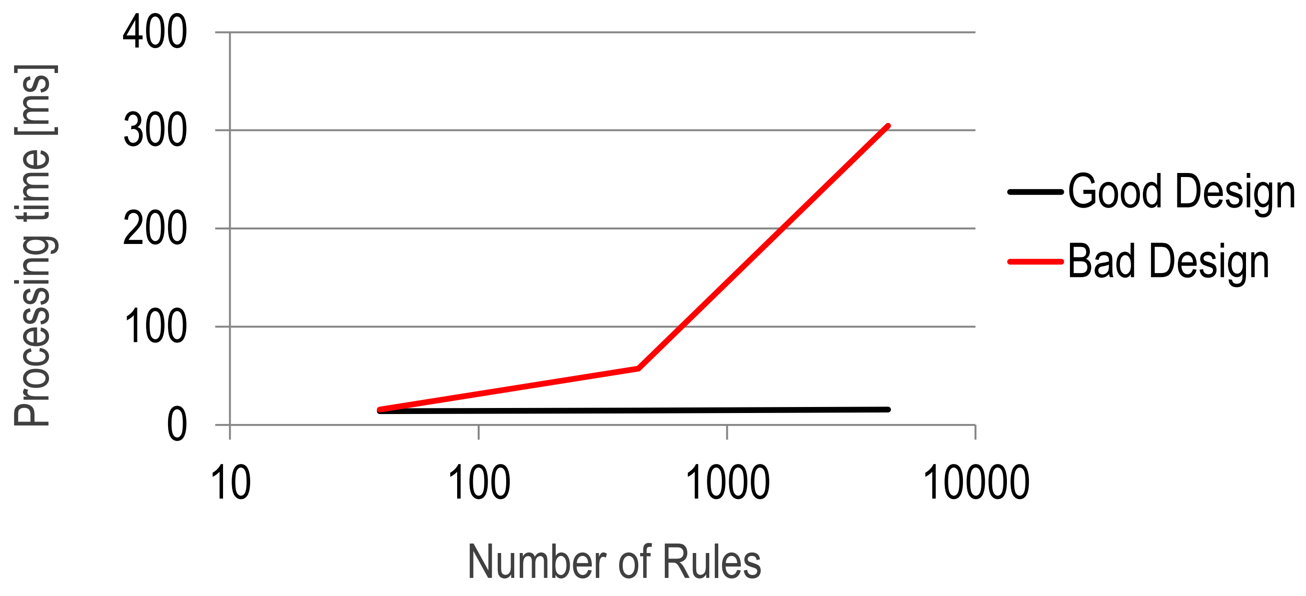
```
<PolicySet PolicyCombiningAlgId="first-applicable">
   <Target/>
   <Policy RuleCombiningAlgId="first-applicable">
      <Target>
         <Match MatchId="function:string-equal">
            <AttributeValue>/users/1/photos</AttributeValue>
            <AttributeDesignator AttributeId="URI" Category="resource" />
         </Match>
      </Target>
      <Rule Effect="Permit">
         <Target>
            <Match MatchId="function:string-equal">
               <AttributeValue>DELETE</AttributeValue>
               <AttributeDesignator AttributeId="HTTP-method" Category="action" />
            </Match>
         </Target>
      </Rule>
   </Policy>
</PolicySet>
```

XACML – Assets & Drawbacks

- Assets
  - Powerful
    - Fine-grained policies
  - Black & White Listing
    - Positive (Permit)
    - Negative (Deny)
  - Technology neutral

- Drawbacks
  - Performance
    - Computation at runtime
    - Bad policy design possible
  - Maintainability
    - Changing policies
    - Error detection/Error resolution
  - Restrictions
    - No overwriting

Policy Design



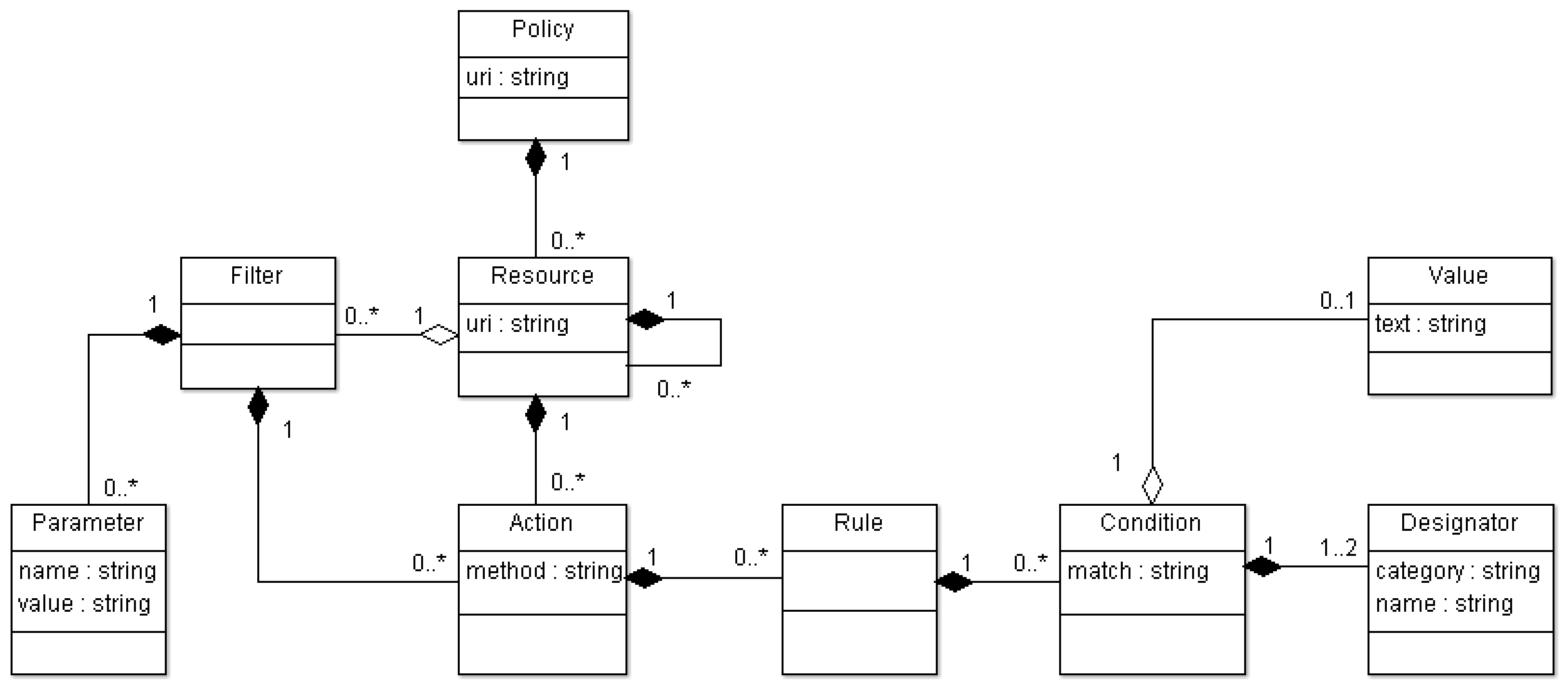
RestACL – Overview

- Inspired by XACML
  - Declarative Authorization
  - Attribute Based
- Goals
  - Avoid performance traps
  - Decrease maintenance efforts
- Intuitive for developers who know REST
  - Resource Oriented
  - Uniform Interface
  - No Targets, no Combining Algorithms

RestACL – Example (XML-Notation)

```
<policy>
  <resource uri="http://example.org">
    <resource uri="/users">
      <action method="DELETE">
        <rule effect="permit" priority="1">
          <condition match="equal">
            <value>192.168.0.0</value>
            <designator category="environment">network</designator>
          </condition>
        </rule>
      </action>
    </resource>
  </resource>
</policy>
```

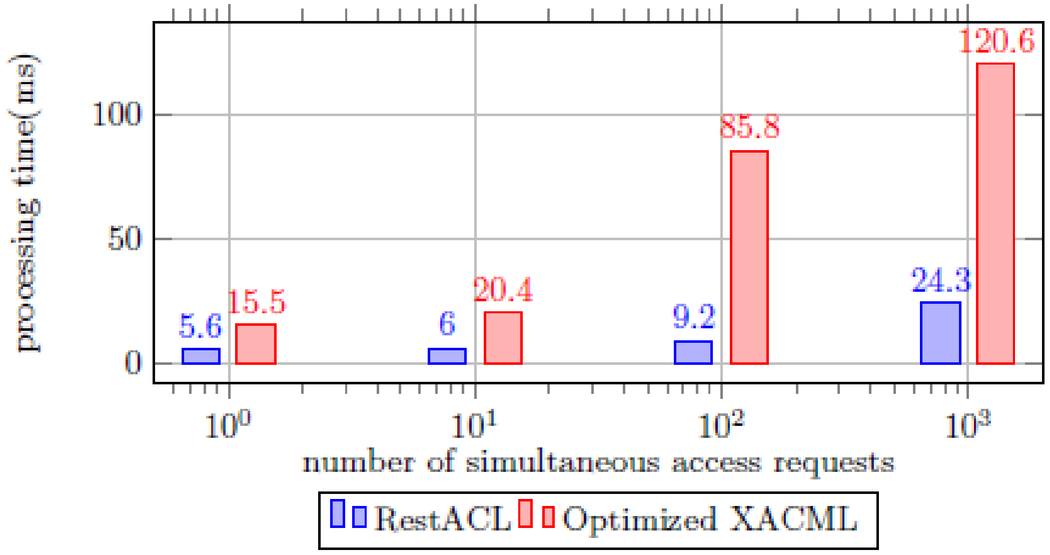
RestACL – Data model



RestACL – Example

```
<policy>
  <resource uri="http://example.org/users">
    <action method="DELETE" id="action1">
      <rule effect="permit" priority="1">
        <condition match="equal">
          <value>192.168.0.0</value>
          <designator category="environment">network</designator>
        </condition>
      </rule>
    </action>
    <resource uri="/1/photos">
      <filter>
        <parameter name="date" value="2015-06-30" />
        <action method="DELETE" or="action1">
          <rule effect="permit" priority="2">
            <condition match="equal">
              <value>huef</value>
              <designator category="subject">id</designator>
            </condition>
          </rule>
        </action>
      </filter>
    </resource>
  </resource>
</policy>
```

Test – RestACL vs. XACML



Test

- RestACL shows the same behaviour like optimized XACML
- RestACL is lightweight
  - Performance benefit
  - Loss of flexibility
    - e.g. Subject oriented policies are not possible

Conclusions

- XACML
  - Powerful
  - Complex
- RestACL
  - Resource Oriented
  - No performance traps
  - Easier to create and maintain Access Rules

Questions