Locating and retrieving policies that use url-match in their target

Background

Suppose we have policies governing access to Web resources. Here's an example …

```
<Policy PolicyId=”p1”>
 <Target>
  <Subjects>
   <Subject>
    <SubjectMatch MatchId=”string-equal”>
     <AttributeValue>customer</AttributeValue>
     <SubjectAttributeDesignator AttributeId=”role”/>
    </SubjectMatch>
   </Subject>
  </Subjects>
  <Resources>
   <Resource>
    <ResourceMatch MatchId=”url-match”>
     <AttributeValue>www.example.com/*</AttributeValue>
     <ResourceAttributeDesignator AttributeId=”resource-id”/>
    </ResourceMatch>
    <ResourceMatch MatchId=”url-match”>
     <AttributeValue>*.cgi</AttributeValue>
     <ResourceAttributeDesignator AttributeId=”resource-id”/>
    </ResourceMatch>
   </Resource>
  </Resources>
  <Actions>
   <Action>
    <ActionMatch MatchId=”string-equal”>
     <AttributeValue>execute</AttributeValue>
     <ActionAttributeDesignator AttributeId=”action-id”/>
    </ActionMatch>
   </Action>
  </Actions>
 </Target>
</Policy>
```

It says that “execute” access to files with the extension “.cgi” on the server “www.example.com/” is to be granted to access-subjects who occupy the role “customer”.

Storage and retrieval using SQL

If we want to be able to store and retrieve such policies by means of an SQL database, then the main database table, Policies, would have the following columns:

Entity (Subject, Resource, Action, Environment)
MatchId
AttributeId

AttributeValue
PolicyId

So, our example policy would be represented in four rows of the Policies table:

Entity	MatchId	AttributeId	AttributeValue	PolicyId
Subject	string-equal	role	customer	p1
Resource	url-match	resource-id	www.example.com/*	p1
Resource	url-match	resource-id	*.cgi	p1
Action	string-equal	action-id	execute	p1

PDP topic

At the time of deployment, a PDP is assigned a "topic". Syntactically, a topic is an XACML <Target>. But, semantically, it defines the set of requests to which the PDP can respond. Upon receiving an access request, the PDP executes all policies applicable to its topic, although certain of those policies may be found not to be applicable to the particular request.

Here is an example of a topic for a PDP that can respond to requests for execute or read access to resources in the folder "www.example.com/resources/" and all of its sub-folders:

```
<Topic>
 <Resources>
  <Resource>
   <ResourceMatch MatchId="url-match">
    <AttributeValue>www.example.com/resources/*</AttributeValue>
    <ResourceAttributeDesignator AttributeId="resource-id"/>
   </ResourceMatch>
  </Resource>
 </Resources>
  <Actions>
   <Action>
    <ActionMatch MatchId="string-equal">
     <AttributeValue>execute</AttributeValue>
     <ActionAttributeDesignator AttributeId="action-id"/>
    </ActionMatch>
   </Action>
   <Action>
    <ActionMatch MatchId="string-equal">
     <AttributeValue>read</AttributeValue>
     <ActionAttributeDesignator AttributeId="action-id"/>
    </ActionMatch>
   </Action>
  </Actions>
</Topic>
```

Our example policy, p1, is applicable to some of the requests that our PDP may have to respond to. So, the PDP should locate, retrieve and load p1.

Using the PDP's topic definition, we have to procedurally generate SQL queries that will retrieve all applicable policies[1]. The applicable policies are all those whose targets intersect with the topic. Or, put another way: applicable policies are those whose targets are identical to, more general than or more specific than the PDP's topic.

The query must therefore be a sequence of select statements for identical, more specific and more general targets. The identical and more specific cases are relatively easy to address: the select statement must include a "where" clause with a wild-card where more specific nodes in the tree would occur. The more general case requires multiple select statements with "where" clauses that truncate the tree structure at successively higher nodes.

file-system-style wild-card

If URL matching uses a simple file-system-style wild-card that can be placed either at the beginning or at the end of a match-string, then it is a straightforward matter to programmatically construct a set of SQL select statements that retrieve all policies applicable to a PDP's topic.

Regular expression

It has been demonstrated that one could express any URL match string using regular expressions. However, regular expressions are much more expressive than this application demands; being capable of expressing patterns that are of no interest whatsoever in URL matching and which will prevent programmatic generation of SQL queries for the purpose of locating and retrieving applicable policies.

Regular expression subset

Maybe there is a subset of the regular expression syntax that (when used in Target) can facilitate this use case. If so, it would not entail a brand new implementation of the regular expression processing logic, leaving out superfluous features, it would merely involve pre-screening expressions and throwing an exception upon encountering one that did not limit itself to the legal subset. Perhaps, the full generality of regular expressions should be allowed in url matches that occur outside the <Target> element.

[1] Note that I'm not talking about doing this when the request is received; I'm talking about doing it at deployment time. So, performance isn't the primary concern.