Technical Lesson 3.1.7

Aggregating Multiple Policies

Goals:

1. Understand how to aggregate multiple **<Policy>** elements in a **<PolicySet>** element.

Summary:

In this Lesson, you will inspect, analyze, and evaluate a policy consisting of a top-level **<PolicySet>** element and multiple **<Policy>** sub-elements. We provide the context of a local implementing agency needing to aggregate policies from multiple levels of authority, illustrating how policy-combining algorithms resolve conflicts among policies in a policy set.

Steps:

3.1.7.1 Inspect PolicySet-1.xml

PolicySet-1 is a **<PolicySet>**. It has a **PolicySetId** identifier (Line 5), and a **PolicyCombiningAlgorithm** of "permit-overrides" (Line 6). Policy-combining algorithms work in a similar manner to rule-combining algorithms. PolicySet-1 specifies a **<Target>** (Lines 16 – 26) and two **<Policy>** elements (the first on Lines 28 – 103 and the second on Lines 105 – 160). The first **<Policy>** represents a federation-level policy (of the "ExampleFederation") and the second represents a policy that's local to the agency that is implementing this **<PolicySet>** ("Agency-A").

This **<PolicySet>** is concerned with access to the criminal history records of Agency-A. Arrest Records constitute the entirety of criminal history data of Agency-A. Accordingly, the **<Target>** of the **<PolicySet>** specifies that the GFIPM Criminal History Data Indicator of the Resource must be true.

The first **<Policy>**, Federation-Policy-1, is the federation-level policy. It can be articulated as: "A federated user can read criminal history data (Arrest Records) if that user meets the following criteria: they are a sworn law enforcement officer, they possess the criminal history data agency home search privilege, and they have legal jurisdiction in the jurisdiction of the record."[1] The Subject *match-predicate* on Lines 53 – 57 uses the "string-regexp-match" XACML function to determine if the Subject is a member of ExampleFederation by checking the GFIPM Federation Id attribute[2].

[1] Note that Federation-Policy-1 duplicates the <**ResourceMatch**> that is in the <**Target**> of the <**PolicySet**> because Federation-Policy-1 needs to be a complete policy in and of itself.
[2] See http://gfipm.net/standards/metadata/2.0/user/FederationId.html.

The second **<Policy>**, Local-Policy-1, is the local-level policy. It can be articulated as: "All sworn law enforcement officers of Agency-A who are authorized to search criminal history data are allowed to read any criminal history record."

PolicySet-1 uses the "permit-overrides" policy combining algorithm, therefore "Permit" decisions take precedence over "Deny" decisions. However, since PolicySet-1 does not contain any "Deny" rules, it will never evaluate to "Deny". It can only evaluate to "Permit" or "NotApplicable"[3].

3.1.7.2 Evaluate PolicySet-1 against Request-1

The **<Target>** of PolicySet-1 will match Request-1 since the request is for criminal history data. Therefore, Federation-Policy-1 will be evaluated.

Federation-Policy-1 will not be applicable to the request since the jurisdiction of the request Subject ("VA") does not match the jurisdiction of the record ("GA"). Therefore, Local-Policy-1 will be evaluated.

Local-Policy-1 will be applicable to the request since the Subject is a member of Agency-A (see Lines 6 – 8 of Request-1), is a sworn law enforcement officer, and is authorized to search criminal history data records (see Lines 24 – 27 of Request-1). Therefore, PolicySet-1 should evaluate to "Permit".

Confirm that the **<Decision>** for Resource-1 is "Permit".

3.1.7.3 Evaluate PolicySet-1 against Request-2

The **<Target>** of PolicySet-1 will match Request-1 since the request is for criminal history data. Therefore, Federation-Policy-1 will be evaluated.

Federation-Policy-1 will not be applicable to the request since the Subject does not have the criminal history data home agency search privilege (see Lines 20 – 23 of Request-2). Therefore, Local-Policy-1 will be evaluated.

Local-Policy-1 will not be applicable to the request since the Subject is not a member of Agency-A (see Lines 6 – 8 of Request-2). Therefore, PolicySet-1 should evaluate to "NotApplicable".

Confirm that the **<Decision>** for Resource-2 is "NotApplicable".

[3] Theoretically, PolicySet-1 can also evaluate to "Indeterminate", however, we have designed the policy and requests to avoid this result.

3.1.7.4 Evaluate PolicySet-1 against Request-3

The **<Target>** of PolicySet-1 will match Request-1 since the request is for criminal history data. Therefore, Federation-Policy-1 will be evaluated.

Federation-Policy-1 will be applicable to the request (you should be able to determine why). Therefore Federation-Policy-1 should evaluate to "Permit". Given the policy-combining algorithm "permit-overrides", there will be no need to evaluate Local-Policy-1, and PolicySet-1 should evaluate to "Permit".

Confirm that the **<Decision>** for Resource-3 is "Permit".

A Note about Notation

XML elements, for XACML and data files, are written as they appear in XML documents, and are indicated in boldface text. For example: **<Policy>**.

XML attributes, for XACML and data files, are written as they appear in XML documents, and are indicated in boldface text. For example: **PolicyId**.

Values of XACML and data elements appear in double quotes. For example: "Permit".

We introduce some terms to serve as labels for certain groups of policy elements; these terms are used to enable discussions about groups of elements as a whole. These terms appear in italics. For example: *class*.

We use labels to refer to files, directories, and data items that exist in the accompanying virtual machine. These labels are used in the style of Linux environment variables – they begin with a dollar sign ($) which is followed by the label in all caps. For example: the label $POLICY_GUIDE refers to the following path on the virtual machine, "/home/guide/policy-guide".