Technical Lesson 3.1.3

Multiple Match-Predicates per Instance, Multiple Instances per Class

Goals:

1. Understand the policy evaluation semantics for multiple *match-predicates* in an *instance*.
2. Understand the policy evaluation semantics for multiple *instances* in a *class*.

Summary:

In this Lesson, you will analyze policies that have multiple *match-predicates* in an *instance* and multiple *instances* in a *class*. You will learn the policy evaluation semantics for both scenarios; multiple *match-predicates* in an *instance* are conjunctive while multiple *instances* in a *class* are disjunctive. You will be challenged to author requests that will achieve certain results.

Steps:

3.1.3.1 Inspect Multiple-Predicate-Policy.xml

Open Multiple-Predicate-Policy.xml. Multiple-Predicate-Policy contains a **<Target>** (Lines 12 - 27) with only the **<Subjects>** *class* specified (Lines 13 – 26). It contains a single **<Rule>**, "Rule-1" (Lines 29 – 37), that has an empty **<Target>** (Line 35) and an **Effect** of "Permit".

The **<Subjects>** *class* of the policy **<Target>** contains a single **<Subject>** *instance* (Lines 14 – 25). This *instance* contains two **<SubjectMatch>** *match-predicate* elements; the first is on Lines 15 – 19, and the second is on Lines 20 – 24. The first *match-predicate* can be read: "The GFIPM Security Clearance Level Code of the Subject is 'Top Secret'." The second *match-predicate* can be read: "The Subject is a Sworn Law Enforcement Officer."

Note that the second *match-predicate* uses a **MatchId** of "boolean-equal". This Function compares two Boolean values for equality. When using the SunXACML library, literal Boolean values (i.e., "true" and "false") must be in lower case.

All *match-predicates* need to evaluate to true for the parent *instance* to match a request (see Table 17: Instance Evaluation Table in the GPPTF Guide's Appendix C for more details). In this policy, requests for which the Subject is a Sworn Law Enforcement Officer with a Top Secret Clearance will match the **<Subject>** *instance*. Since this is the only *instance* in the policy, and the policy has a single rule, then this policy should evaluate to "Permit" for Sworn Law Enforcement Officers who have a Top Secret Clearance performing any action to any resource in any environment.

3.1.3.2 Inspect Multiple-Instance-Policy.xml

Open Multiple-Instance-Policy.xml. Multiple-Instance-Policy contains a **<Target>** (Lines 12 - 29) with only the **<Subjects>** *class* specified (Lines 13 – 28). It contains a single **<Rule>**, "Rule-1" (Lines 31 – 39), that has an empty **<Target>** (Line 37) and an **Effect** of "Permit".

The **<Subjects>** *class* of the policy **<Target>** contains two **<Subject>** *instances*. The first is on Lines 14 – 20, and the second is on Lines 21 – 27. Each **<Subject>** *instance* contains a single **<SubjectMatch>** *match-predicate*.

The **<SubjectMatch>** of the first **<Subject>** (Lines 15 – 20) also exists in Multiple-Predicate-Policy. It can be read: "The GFIPM Security Clearance Level Code of the Subject is 'Top Secret'."

The **<SubjectMatch>** *match-predicate* of the second **<Subject>** *instance* (Lines 22 – 26) also exists in Multiple-Predicate-Policy. It can be read: "The Subject is a Sworn Law Enforcement Officer."

For a *class* to match a request, at least one of its *instances* must match the request (see Table 18: Class Evaluation Table in the GPPTF Guide's Appendix C for more details). For this policy, the **<Subjects>** *class* will match requests for which the Subject either has a Top Secret Clearance, or is a Sworn Law Enforcement Officer, or both. Since the **<Subjects>** *class* is the only *class* specified, and there is only one rule, this policy will evaluate to "Permit" for requests that its **<Subjects>** *class* matches.

3.1.3.3 Compare Multiple-Predicate-Policy to Multiple-Instance-Policy

These policies both include the same *match-predicates*. However, since Multiple-Predicate-Policy organizes the *match-predicates* within the same *instance*, and Multiple-Instance-Policy organizes the *match-predicates* in separate *instances*, the semantics of these two policies are different (as described in Steps 0 and 0). Multiple-Predicate-Policy is more restrictive since both *match-predicates* must evaluate to true for that policy to be applicable to a request. Also, Multiple-Instance-Policy will be applicable to every request to which Multiple-Predicate-Policy is applicable.

3.1.3.4 Challenge: Create Request-1

Create a new XML file called "Request-1.xml". In this file, author a request that will be applicable to Multiple-Predicate-Policy. Because of how the two policies are written, this request should also be applicable to Multiple-Instance-Policy. The request should include Subject attributes, and a "resource-id" Resource attribute. For the "resource-id" attribute, use a value of "Resource-1". You can leave the Action and Environment sections empty.

A solution to this Challenge is in Request-1-Solution.xml.

3.1.3.5 Evaluate Multiple-Predicate-Policy against your Request-1

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

3.1.3.6 Evaluate Multiple-Instance-Policy against your Request-1

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

3.1.3.7 Challenge: Create Request-2

Create a new XML file called “Request-2.xml”. In this file, author a request that will not be applicable to Multiple-Predicate-Policy, but will be applicable to Multiple-Instance-Policy. The request should include Subject attributes, and a “resource-id” Resource attribute. For the “resource-id” attribute, use a value of “Resource-1”. You can leave the Action and Environment sections empty.

A solution to this Challenge is in Request-2-Solution.xml.

3.1.3.8 Evaluate Multiple-Predicate-Policy against your Request-2

Confirm that the **<Decision>** for “Resource-1” is “NotApplicable”.

3.1.3.9 Evaluate Multiple-Instance-Policy against your Request-2

Confirm that the **<Decision>** for “Resource-1” 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”.