University Policy Test Cases

(InfoBeyond Technology LLC)

Abstract

This document demonstrates access control test cases using Security Policy Tool, a software tool for Access Control Security Managers, Policy Authors, and other IT Security Professionals specializing in the performance of access control systems. Access control policies are designed to protect the accessibility of online resources in networks, IoTs, healthcare systems, financial service systems, enterprise IT and clouds, military systems, and other online environments. There are several challenges in building robust access control models for these systems including (i) effectively composing secure policies and rules, (ii) testing these policies systematically, (iii) verifying these policies to prevent access control leaks. Security Policy Tool solves these issues by providing powerful access control policy modeling, testing, and verification features that empower organizations to close the door to access control leaks.

Index Terms

Access control, attribute-based access control (abac), role-based access control (rbac), security policy editing, test, verification, deployment, access control leaks, XACML, software tool.

1 Introduction to Test Cases

This document and linked Security Policy Tool– Project Files have been designed to help you gain an understanding of what common access control errors look like, how they are created, and how to resolve them. Organizations who leverage Security Policy Tool's systematic modeling, testing and verification features are empowered to efficiently identify errors and close the door to access control leaks.

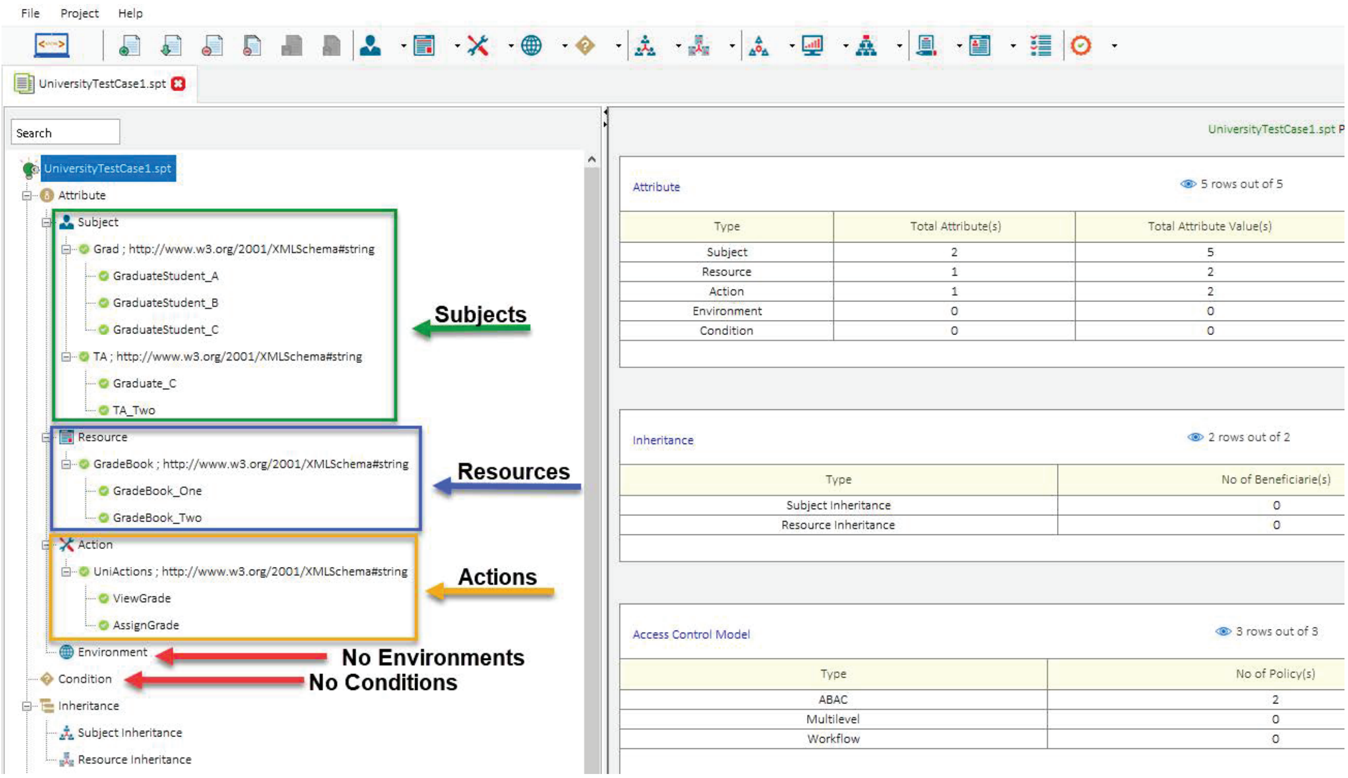
These University Policy test cases are based on examples previously created by the National Institute of Standards & Technology (NIST) to demonstrate commonly found errors in access control policy logic similarly. These test cases consist of policies/rules from NIST's example as well as modifications to better illustrate how Security Policy Tool enhances access control security. The goal of these test cases is to provide a starting point for what to expect as you go on to use Security Policy Tool to analyze your own policy verification results for errors.

2 Setting Up The Policies – Test Case 1 (Rule Conflict)

This university example contains two policies (GradePolicy & TAPolicy). The Attribute /Attribute Values included in these policies are as shown in Figure 1.

- *Contact us at: E-mail: Info@Securitypolicytool.com*

Security Policy Tool (www.Securitypolicytool.com) is a commercial version of NIST(National Institute of Standards and Technology)'s ACPT (Access Control Policy Tool). With tremendous consultation with NIST experts, Security Policy Tool substantially enhances and expands the NIST's ACPT design with advanced features for achieving high security confidence access control levels such that it can be commercialized. The development of Security Policy Tool is financially sponsored by NIST via a SBIR (Small Business Innovation Research) Phase I and II programs. It specifically improves the NIST's ACPT design to provide a robust, unified, professional, and functionally powerful access control policy tool.



UniversityTestCase1.spt
- Attribute
  - Subject
    - Grad ; http://www.w3.org/2001/XMLSchema#string
      - GraduateStudent_A
      - GraduateStudent_B
      - GraduateStudent_C
    - TA ; http://www.w3.org/2001/XMLSchema#string
      - Graduate_C
      - TA_Two
  - Resource
    - GradeBook ; http://www.w3.org/2001/XMLSchema#string
      - GradeBook_One
      - GradeBook_Two
  - Action
    - UniActions ; http://www.w3.org/2001/XMLSchema#string
      - ViewGrade
      - AssignGrade
  - Environment
- Condition
- Inheritance
  - Subject Inheritance
  - Resource Inheritance

UniversityTestCase1.spt P

Attribute — 5 rows out of 5

Type	Total Attribute(s)	Total Attribute Value(s)
Subject	2	5
Resource	1	2
Action	1	2
Environment	0	0
Condition	0	0

Inheritance — 2 rows out of 2

Type	No of Beneficiarie(s)
Subject Inheritance	0
Resource Inheritance	0

Access Control Model — 3 rows out of 3

Type	No of Policy(s)
ABAC	2
Multilevel	0
Workflow	0

Fig. 1. Test Case 1

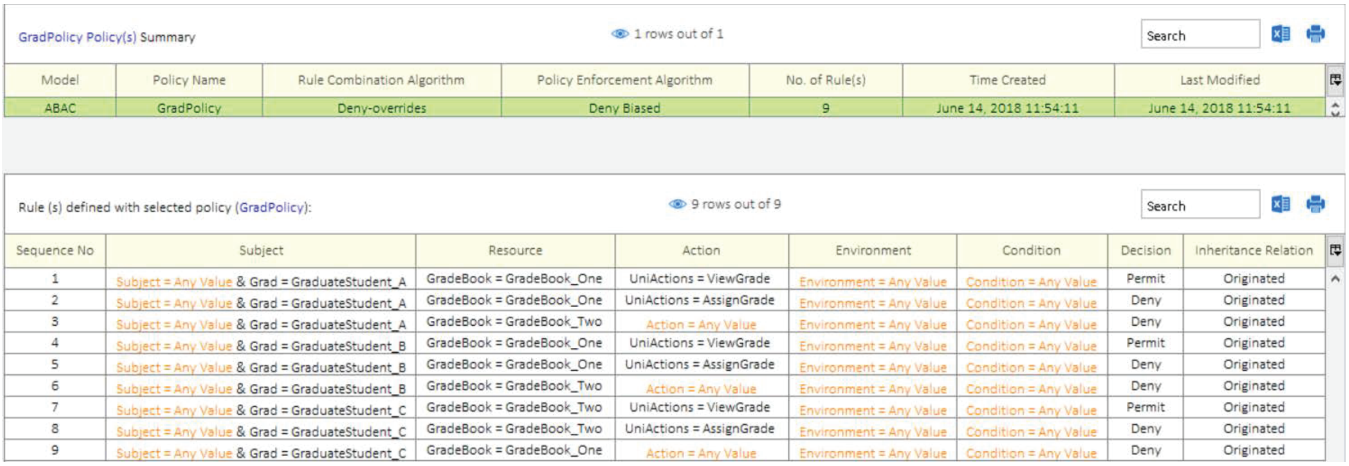
3 Modeling Your Policy – Test Case 1 (Rule Conflict)

Now that we have entered our attributes we can model our two policies (GradePolicy & TAPolicy). See the list below of the rules contained in each of these policies. You can open a "New (blank) Project" and build these policies by entering the following rules below:

GradePolicy:
(Subject = Any Value & Grad = GraduateStudent_A, ViewGrade, GradeBook_One) → Permit
(Subject = Any Value & Grad = GraduateStudent_A, AssignGrade, GradeBook_One) →Deny
(Subject = Any Value & Grad = GraduateStudent_A, Any Action, GradeBook_Two) →Deny
(Subject = Any Value & Grad = GraduateStudent_B, ViewGrade, GradeBook_One) →Permit
(Subject = Any Value & Grad = GraduateStudent_B, AssignGrade, GradeBook_One) →Deny
(Subject = Any Value & Grad = GraduateStudent_B, Any Action, GradeBook_Two) →Deny
(Subject = Any Value & Grad = GraduateStudent_C, ViewGrade, GradeBook_Two) →Permit
(Subject = Any Value & Grad = GraduateStudent_C, AssignGrade, GradeBook_Two) →Deny
(Subject = Any Value & Grad = GraduateStudent_C, Any Action, GradeBook_One) →Deny

TAPolicy:
(Subject = Any Value & TA = GraduateStudent_C, ViewGrade, GradeBook_One) →Permit
(Subject = Any Value & TA = GraduateStudent_C, AssignGrade, GradeBook_One) →Deny
(Subject = Any Value & TA = GraduateStudent_C, Any Action, GradeBook_Two) →Deny
(Subject = Any Value & TA = TA_Two, ViewGrade, GradeBook_Two) →Permit
(Subject = Any Value & TA = TA_Two, AssignGrade, GradeBook_Two) →Permit
(Subject = Any Value & TA = TA_Two, Any Action, GradeBook_One) →Deny

After entering the rules above your modeled policies should look like the screenshots below. If you did not create your own Project File, you can simply open Security Policy Tool – Project File: UniversityTestCase1 and these policies will have been already created for you.



GradPolicy Policy(s) Summary

1 rows out of 1

Search

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	GradPolicy	Deny-overrides	Deny Biased	9	June 14, 2018 11:54:11	June 14, 2018 11:54:11

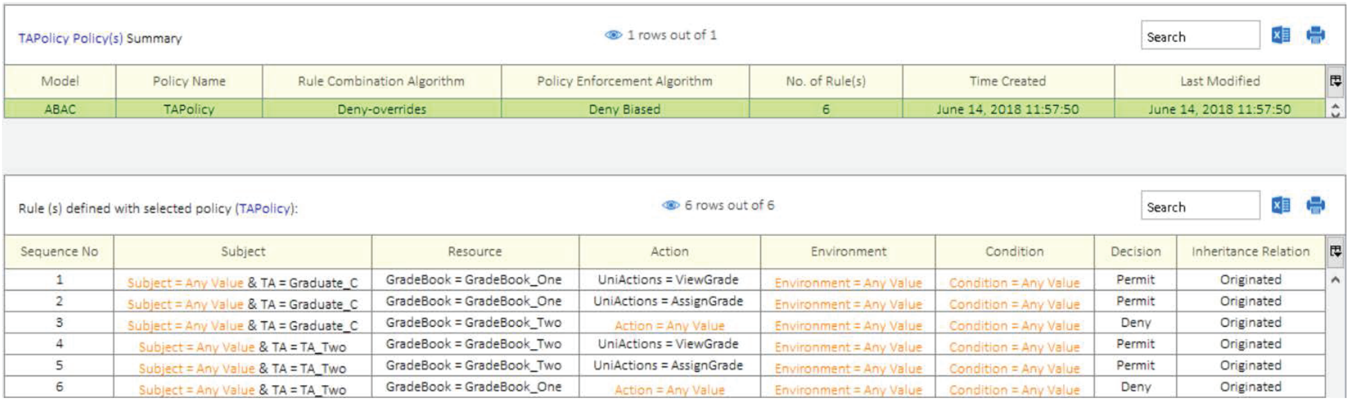
Rule (s) defined with selected policy (GradPolicy):

9 rows out of 9

Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
3	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
6	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
7	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
8	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
9	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 2. GradePolicy



TAPolicy Policy(s) Summary

1 rows out of 1

Search

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	TAPolicy	Deny-overrides	Deny Biased	6	June 14, 2018 11:57:50	June 14, 2018 11:57:50

Rule (s) defined with selected policy (TAPolicy):

6 rows out of 6

Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
3	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
6	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 3. TAPolicy

4 Individual Security Requirements - Test Case 1 (Rule Conflict)

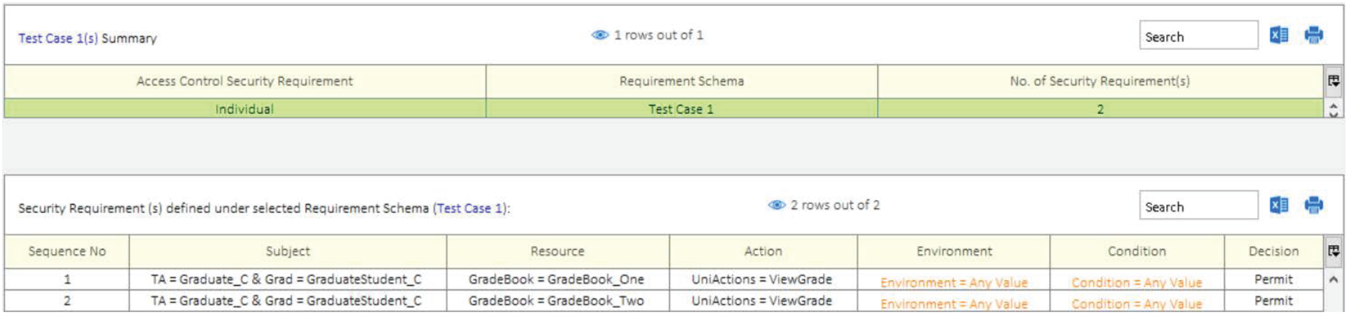
The final step before analyzing these policies for errors is to create individual security requirements to use for testing. If you are building a "New (blank) Project" on your own you will enter the security requirements as follows.

Individual Security Requirements:

(Grad = GraduateStudent_C & TA = GraduateStudent_C) & (Action = ViewGrade) & (GradeBook = GradeBook_One) → decision = Permit

(Grad = GraduateStudent_C & TA = GraduateStudent_C) & (Action = ViewGrade) & (GradeBook = GradeBook_Two) → decision = Permit

After entering the rules above your individual security requirements should look like the screenshots below. If you did not create your own Project File you can simply open Security Policy Tool – Project File: UniversityTestCase1 and these requirements will have been already created for you.



Test Case 1(s) Summary — 1 rows out of 1 — Search

Access Control Security Requirement	Requirement Schema	No. of Security Requirement(s)
Individual	Test Case 1	2

Security Requirement (s) defined under selected Requirement Schema (Test Case 1): — 2 rows out of 2 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision
1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit
2	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit

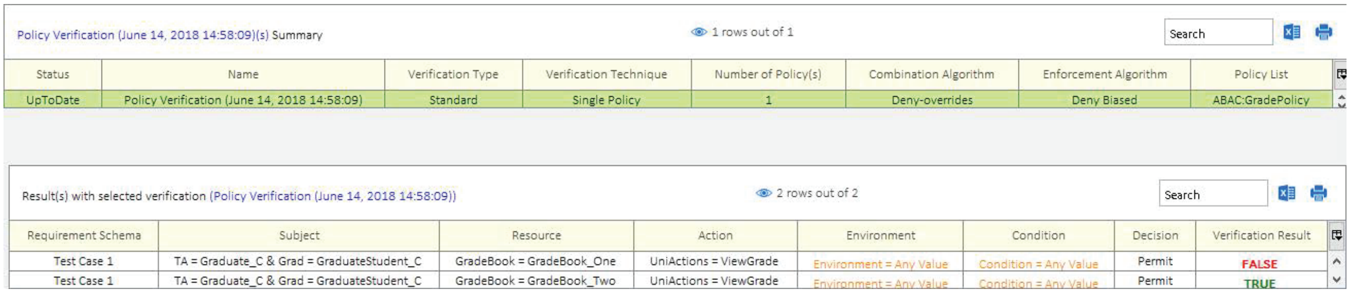
Fig. 4. Individual Security Requirements

5 Policy Verification/Analyzing Results - Test Case 1 (Rule Conflict)

Now that we are ready to test our policies let's discuss the error we will be looking at in this first example. When policies are designed, there is potential for a "Rule Conflict" being created. A Rule Conflict occurs when two or more rules are defining opposite authorization in an access control policy.

In our example, an individual at this university has a role of both TA and Graduate Student at the facility. Due to this, the individual is assigned both (TA: GraduateStudent_C and Grad: GraduateStudent_C) attribute values by the system during access evaluation. In the GradePolicy it defines that graduate students can view grades but cannot assign grades. However, in the TAPolicy graduate students can assign grades resulting in a Rule Conflict (e.g., Permitted to AssignGrade in TAPolicy, Denied to AssignGrade in GradePolicy).

Next, we will run two "Single Policy" Verifications to reveal the Rule Conflict that is present in our policies. To do this, we will select GradePolicy and Test Case 1 (security requirement) as a Single Policy Verification and also choose TAPolicy and Test Case 1 (security requirement) as a Single Policy Verification and analyze our verification results. Again, this will have already been done for you if you open Project File: UniversityTestCase1.



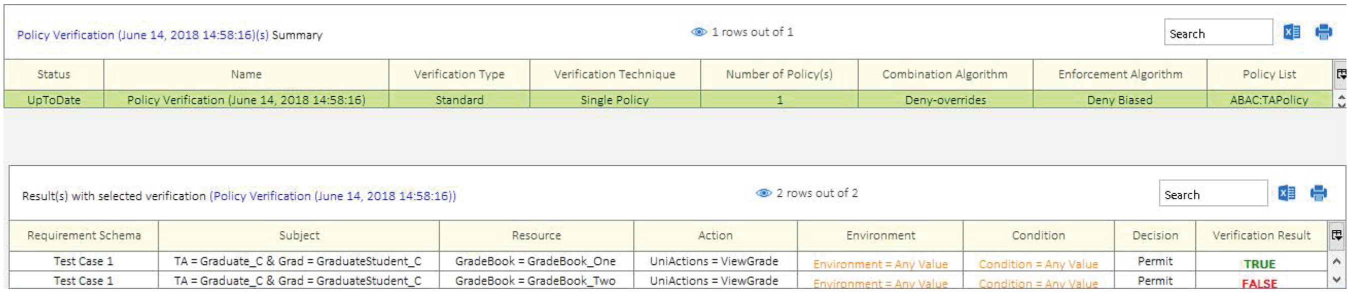
Policy Verification (June 14, 2018 14:58:09)(s) Summary — 1 rows out of 1 — Search

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 14:58:09)	Standard	Single Policy	1	Deny-overrides	Deny Biased	ABAC:GradePolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:09)) — 2 rows out of 2 — Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE

Fig. 5. GradePolicy x Test Case 1



Policy Verification (June 14, 2018 14:58:16)(s) Summary — 1 rows out of 1 — Search

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 14:58:16)	Standard	Single Policy	1	Deny-overrides	Deny Biased	ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:16)) — 2 rows out of 2 — Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

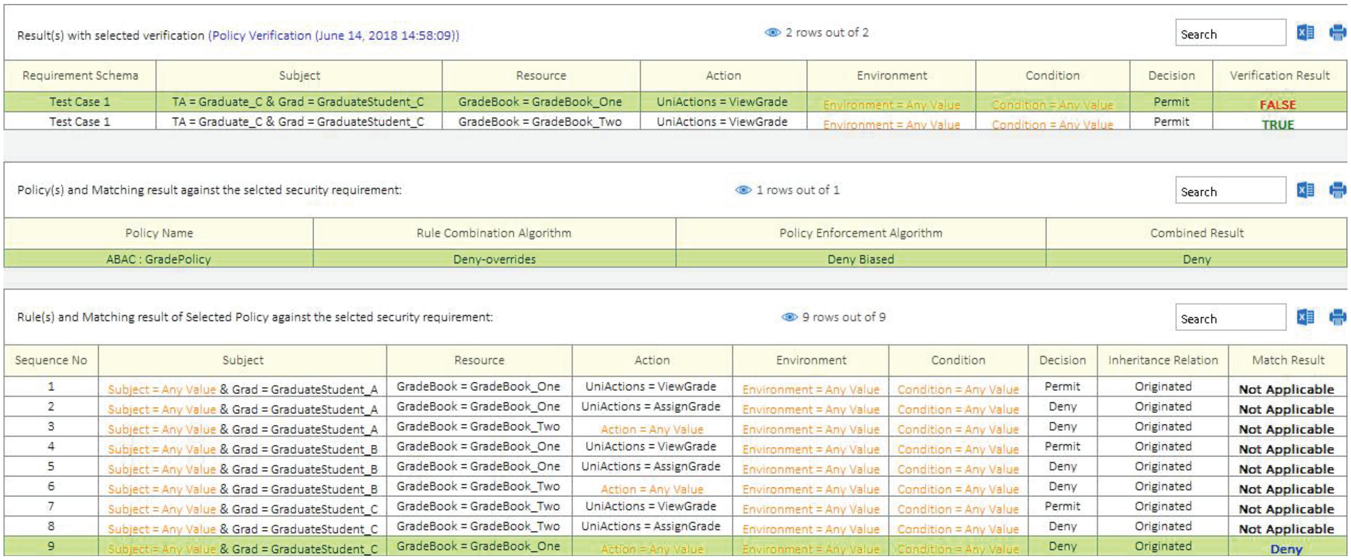
Fig. 6. TAPolicy x Test Case 1

As you can see from our verification results our policies are both Permitting and Denying the individual (Grad = GraduateStudent_C/TA = GraduateStudent_C) from viewing GradeBook_One and GradeBook_Two which is known as a Rule Conflict error.

6 Resolving This Error - Test Case 1 (Rule Conflict)

To solve a Rule Conflict the policy author would need to go back and either update or delete the related rules to the error. To view which specific Rules are resulting in these Verification Results we can click on all (4) of our specific Results (GradePolicyxTestCase1: False;True & TAPolicyxTestCase1: True;False) and see which Rules have "Match Results".

See the screenshots below of our two Policies Match Results to discover which specific rules are related to our Verification Results (e.g., False, True).



Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:09)) 2 rows out of 2 Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE

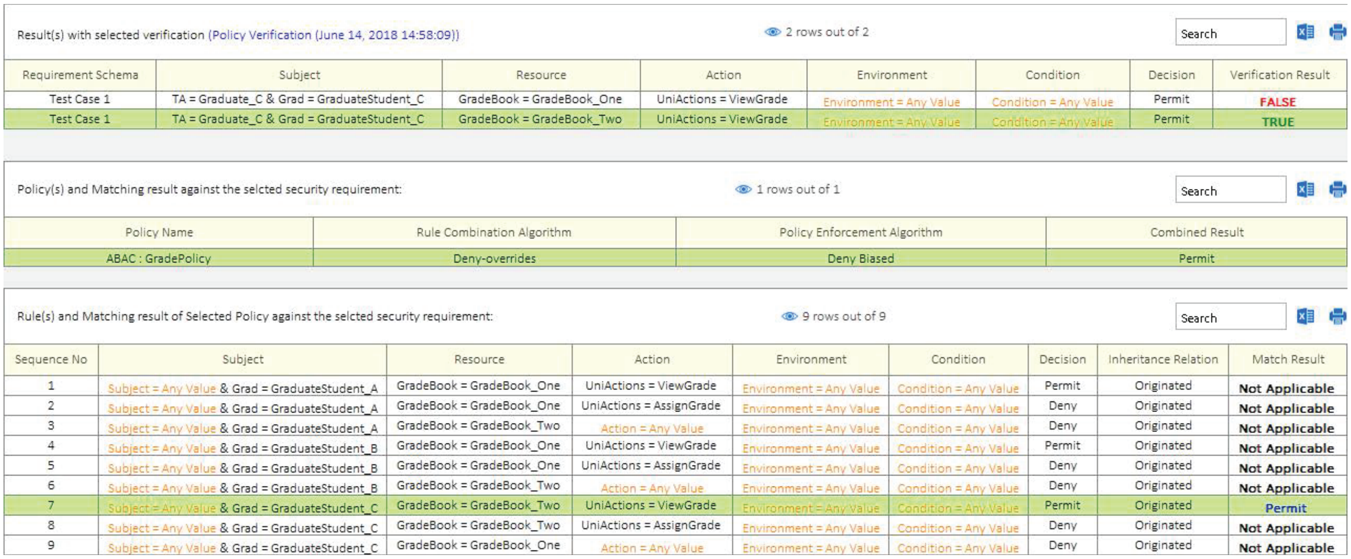
Policy(s) and Matching result against the selcted security requirement: 1 rows out of 1 Search

Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement: 9 rows out of 9 Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
2	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
3	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
4	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
5	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
6	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
7	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
8	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
9	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Deny

Fig. 7. GradePolicy: Match Results (GradeBook_One)



Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:09)) 2 rows out of 2 Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE

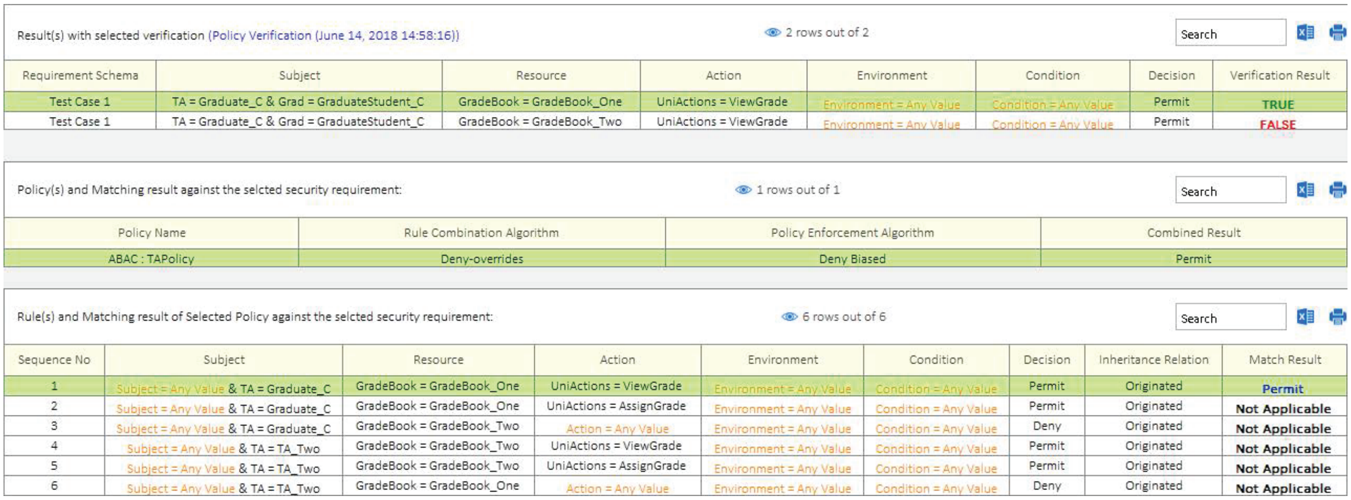
Policy(s) and Matching result against the selcted security requirement: 1 rows out of 1 Search

Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
ABAC : GradePolicy	Deny-overrides	Deny Biased	Permit

Rule(s) and Matching result of Selected Policy against the selcted security requirement: 9 rows out of 9 Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
2	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
3	Subject = Any Value & Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
4	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
5	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
6	Subject = Any Value & Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
7	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Permit
8	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
9	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable

Fig. 8. GradePolicy: Match Results (GradeBook_Two)



Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:16)) 2 rows out of 2 Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

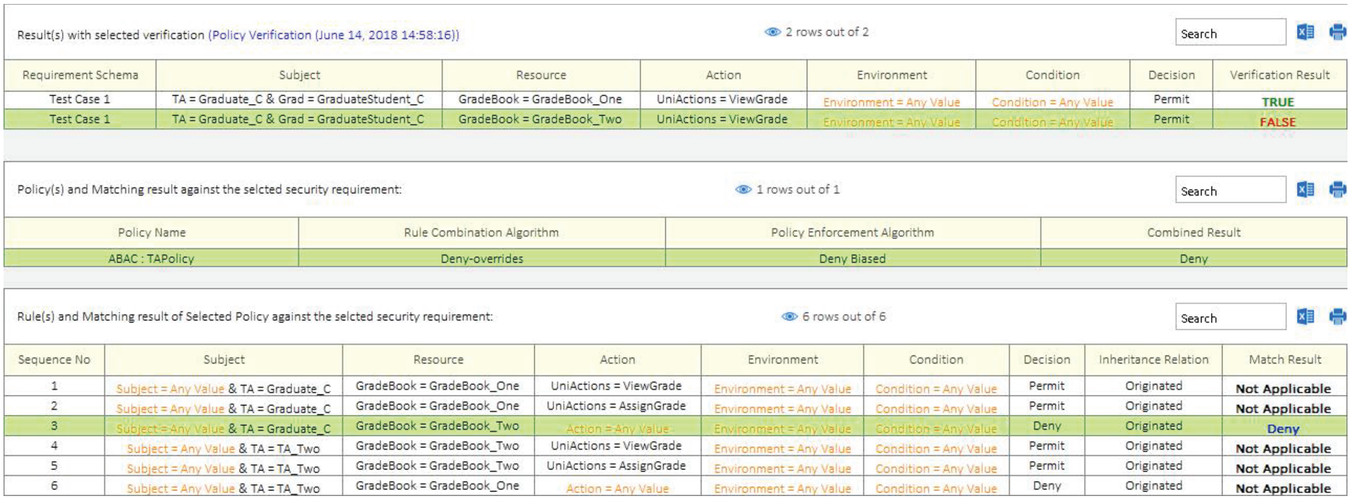
Policy(s) and Matching result against the selcted security requirement: 1 rows out of 1 Search

Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
ABAC : TAPolicy	Deny-overrides	Deny Biased	Permit

Rule(s) and Matching result of Selected Policy against the selcted security requirement: 6 rows out of 6 Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Permit
2	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
3	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
4	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
5	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
6	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable

Fig. 9. TAPolicy: Match Results (GradeBook_One)



Result(s) with selected verification (Policy Verification (June 14, 2018 14:58:16)) 2 rows out of 2 Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement: 1 rows out of 1 Search

Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
ABAC : TAPolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement: 6 rows out of 6 Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
2	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
3	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Deny
4	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
5	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
6	Subject = Any Value & TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable

Fig. 10. TAPolicy: Match Results (GradeBook_Two)

Now that we have pinpointed our (4) Rules related to our Rule Conflict Error we can go back and make changes or possibly remove these rules. Depending on your organizational structure the policy author or access control administrator would need to decide what is the most appropriate action to take to resolve the error. There is no “right” or “wrong” solution for this, you would need to determine what is suitable based on your organizational needs.

For our example, let’s assume this individual within both of our policies (GraduateStudent_C) should be allowed to View GradeBook_Two & GradeBook_One, however, the individual should NOT be able to AssignGrade to GradeBook_Two because within GradeBook_Two is his own grade. He can still AssignGrade to GradeBook_One because it is required for his TA duties. To resolve this, we will add 2 rules and delete 1 rule in both policies which will in turn resolve the Rule Conflict.

GradePolicy: Delete (1) Current Rule:

(Rule No. = 9) → (Subject = Any Value & Grad = GraduateStudent_C) → (Action = Any Value) → (Resource = GradeBook_One) → decision = Deny

GradePolicy: Add (2) New Rules:

(Rule No. = 9) → (Subject = Any Value & Grad = GraduateStudent_C) → (Action = ViewGrade) → (Resource = GradeBook_One) → decision = Permit

(Rule No. = 10) → (Subject = Any Value & Grad = GraduateStudent_C) → (Action = AssignGrade) → (Resource = GradeBook_One) → decision = Permit

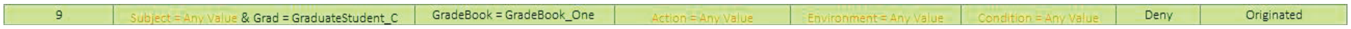
TAPolicy: Delete (1) Current Rule:

(Rule No. = 3) → (Subject = Any Value & TA = GraduateStudent_C) → (Action = Any Value) → (Resource = GradeBook_Two) → decision = Deny

TAPolicy: Add (2) New Rules:

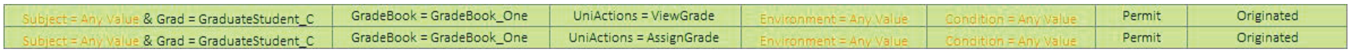
(Rule No. = 6) → (Subject = Any Value & TA = GraduateStudent_C) → (Action = ViewGrade) → (Resource = GradeBook_Two) → decision = Permit

(Rule No. = 7) → (Subject = Any Value & TA = GraduateStudent_C) → (Action = AssignGrade) → (Resource = GradeBook_Two) → decision = Permit



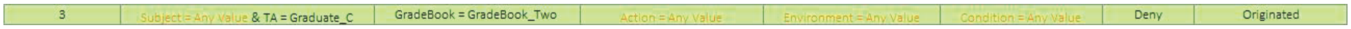
9	Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 11. GradePolicy: Delete Rule (9)



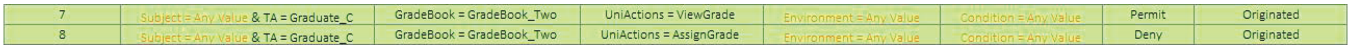
Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
Subject = Any Value & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated

Fig. 12. GradePolicy: Add New Rules



3	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

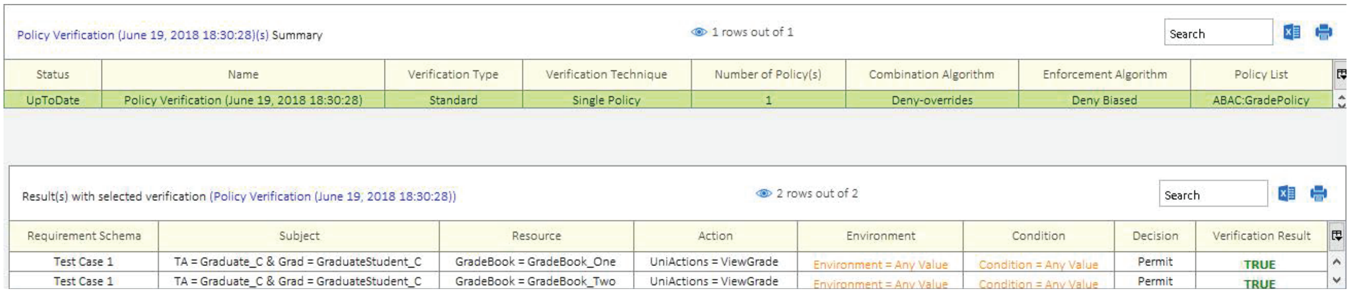
Fig. 13. TAPolicy: Delete Rule (3)



7	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
8	Subject = Any Value & TA = Graduate_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 14. TAPolicy: Add New Rules

After we “Refresh” our previous Verification Results we no longer have a Rule Conflict occurring. . . .



Policy Verification (June 19, 2018 18:30:28)(s) Summary

1 rows out of 1

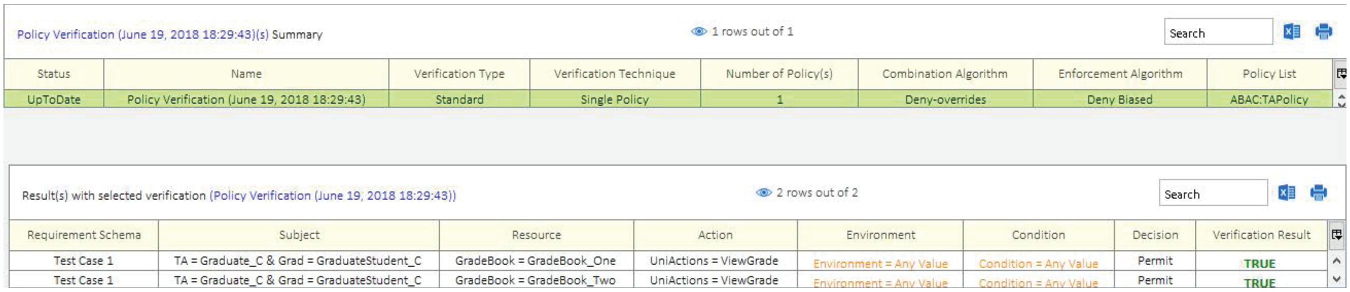
Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 19, 2018 18:30:28)	Standard	Single Policy	1	Deny-overrides	Deny Biased	ABAC:GradePolicy

Result(s) with selected verification (Policy Verification (June 19, 2018 18:30:28))

2 rows out of 2

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE

Fig. 15. Updated Results: GradePolicy (No Rule Conflict)



Policy Verification (June 19, 2018 18:29:43)(s) Summary

1 rows out of 1

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 19, 2018 18:29:43)	Standard	Single Policy	1	Deny-overrides	Deny Biased	ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 19, 2018 18:29:43))

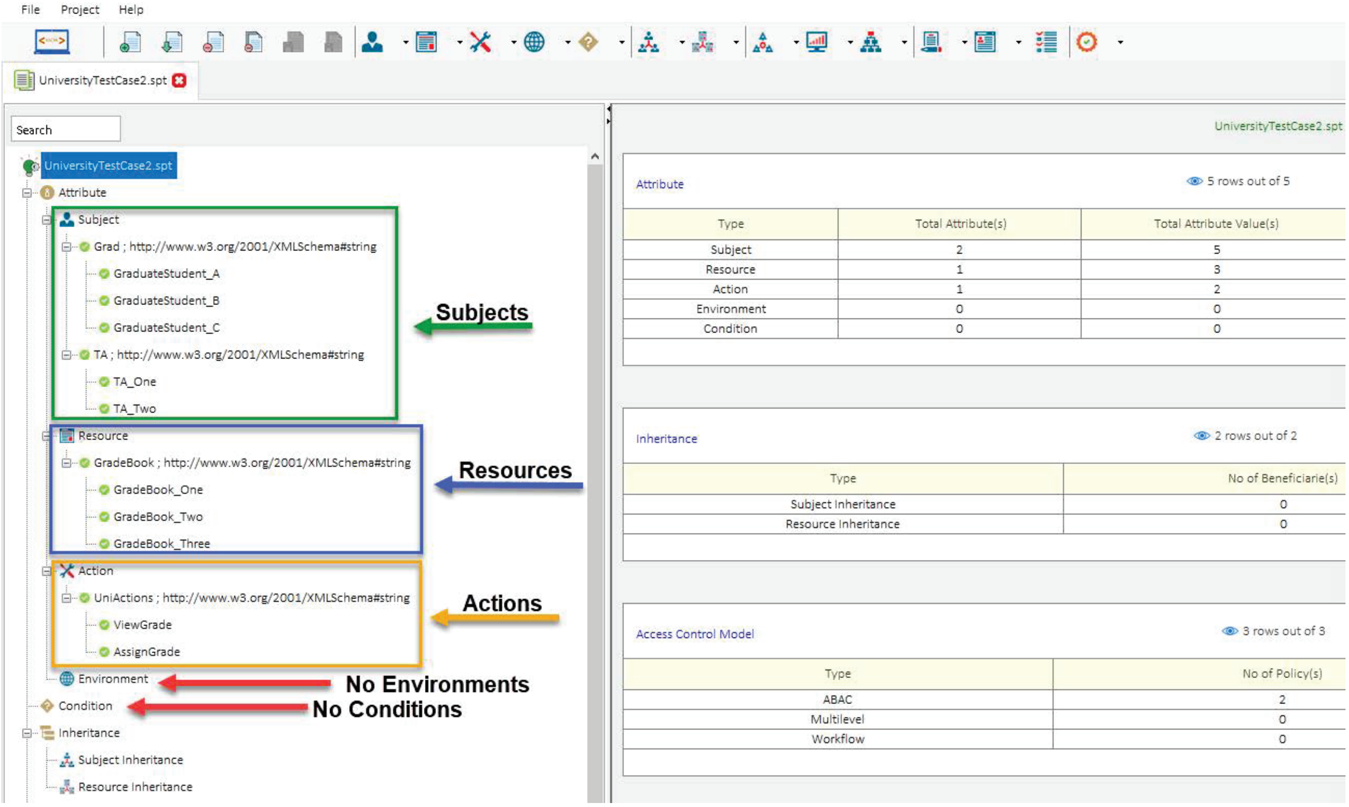
2 rows out of 2

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE
Test Case 1	TA = Graduate_C & Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	TRUE

Fig. 16. Updated Results: TAPolicy (No Rule Conflict)

7 Setting Up The Policies – Test Case 2 (Not Protected Resource)

This university example contains two policies (GradePolicy & TAPolicy). The attributes in this example have been changed slightly from previous Test Case 1. TA’s attribute value has been changed from “GraduateStudent_C” to “TA_One” and also GradeBook has gained a new attribute value called “GradeBook_Three.” The Attribute/Attribute Values included in these policies are as shown in Figure 17.



Attribute — 5 rows out of 5

Type	Total Attribute(s)	Total Attribute Value(s)
Subject	2	5
Resource	1	3
Action	1	2
Environment	0	0
Condition	0	0

Inheritance — 2 rows out of 2

Type	No of Beneficiarie(s)
Subject Inheritance	0
Resource Inheritance	0

Access Control Model — 3 rows out of 3

Type	No of Policy(s)
ABAC	2
Multilevel	0
Workflow	0

Fig. 17. Test Case 2

8 Modeling Your Policy – Test Case 2 (Not Protected Resource)

Now that we have entered our attributes we can model our two policies (GradePolicy & TAPolicy). See the list below of the rules contained in each of these policies. You can open a “New (blank) Project” and build these policies by entering the following rules below:

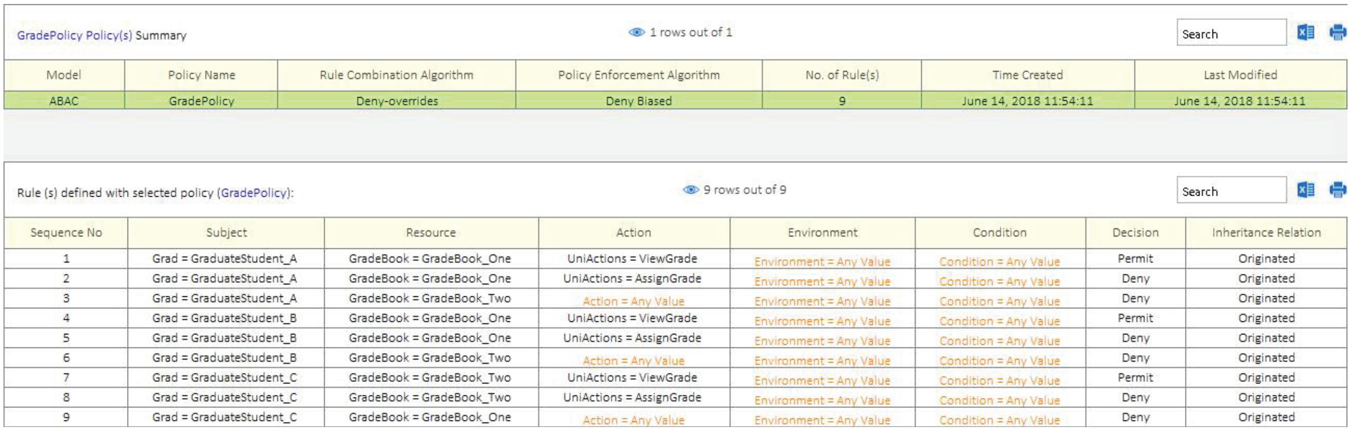
GradePolicy:

(Grad = GraduateStudent_A, ViewGrade, GradeBook_One) → Permit
(Grad = GraduateStudent_A, AssignGrade, GradeBook_One) →Deny
(Grad = GraduateStudent_A, Any Action, GradeBook_Two) →Deny
(Grad = GraduateStudent_B, ViewGrade, GradeBook_One) →Permit
(Grad = GraduateStudent_B, AssignGrade, GradeBook_One) →Deny
(Grad = GraduateStudent_B, Any Action, GradeBook_Two) →Deny
(Grad = GraduateStudent_C, ViewGrade, GradeBook_Two) →Permit
(Grad = GraduateStudent_C, AssignGrade, GradeBook_Two) →Deny
(Grad = GraduateStudent_C, Any Action, GradeBook_One) →Deny

TAPolicy:

(TA = TA_One, ViewGrade, GradeBook_One) →Permit
(TA = TA_One, AssignGrade, GradeBook_One) →Permit
(TA = TA_One, Any Action, GradeBook_Two) →Deny
(TA = TA_Two, ViewGrade, GradeBook_Two) →Permit
(TA = TA_Two, AssignGrade, GradeBook_Two) →Permit
(TA = TA_Two, Any Action, GradeBook_One) →Deny

After entering the rules above your modeled policies should look like the screenshots below. If you did not create your own Project File, you can simply open Security Policy Tool – Project File: UniversityTestCase2 and these policies will have been already created for you.



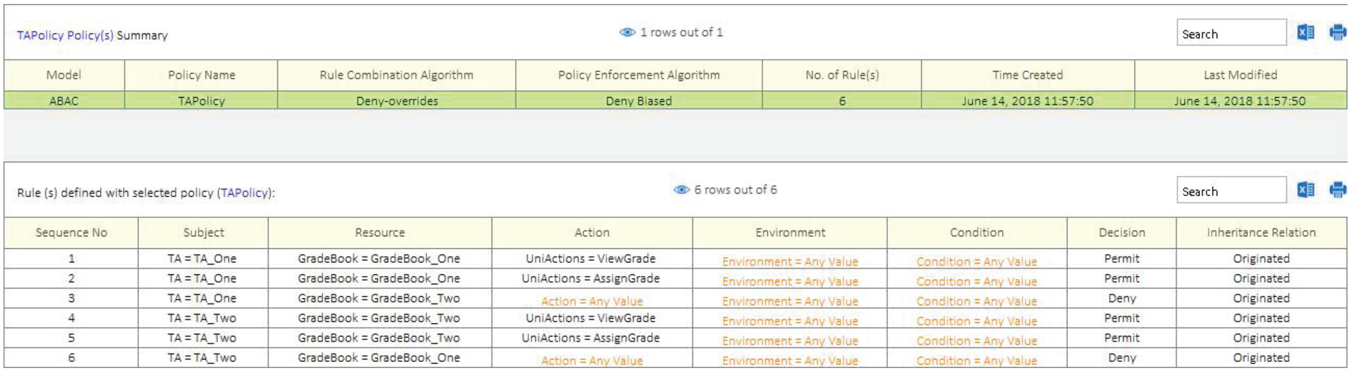
GradePolicy Policy(s) Summary — 1 rows out of 1 — Search

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	GradePolicy	Deny-overrides	Deny Biased	9	June 14, 2018 11:54:11	June 14, 2018 11:54:11

Rule (s) defined with selected policy (GradePolicy): — 9 rows out of 9 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
3	Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
6	Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
7	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
8	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
9	Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 18. GradePolicy



TAPolicy Policy(s) Summary — 1 rows out of 1 — Search

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	TAPolicy	Deny-overrides	Deny Biased	6	June 14, 2018 11:57:50	June 14, 2018 11:57:50

Rule (s) defined with selected policy (TAPolicy): — 6 rows out of 6 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 19. TAPolicy

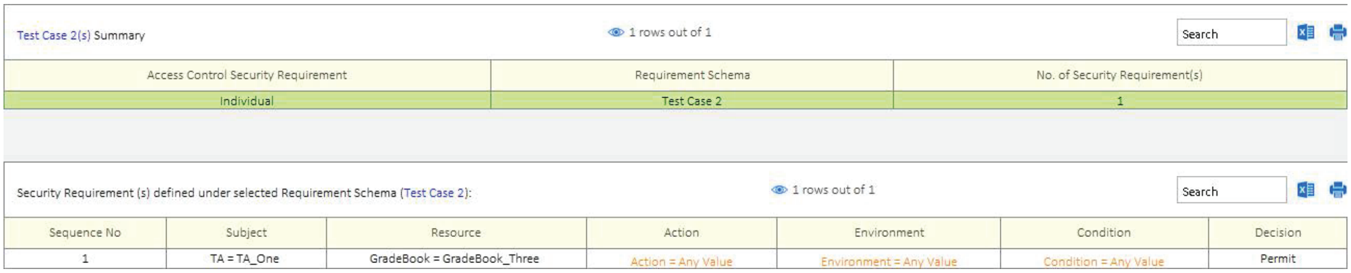
9 Individual Security Requirements - Test Case 2 (Not Protected Resource)

The final step before analyzing these policies for errors is to create individual security requirements to use for testing. If you are building a "New (blank) Project" on your own you will enter the following security requirement below:

Individual Security Requirement:

(TA = TA_One) & (Action = Any) & (GradeBook =GradeBook_Three) → decision = Permit

After entering the rule above your individual security requirement should look like the screenshot below. If you did not create your own Project File you can simply open Security Policy Tool – Project File: UniversityTestCase2 and this requirement will have been already created for you.



Test Case 2(s) Summary — 1 rows out of 1 — Search

Access Control Security Requirement	Requirement Schema	No. of Security Requirement(s)
Individual	Test Case 2	1

Security Requirement (s) defined under selected Requirement Schema (Test Case 2): — 1 rows out of 1 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision
1	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit

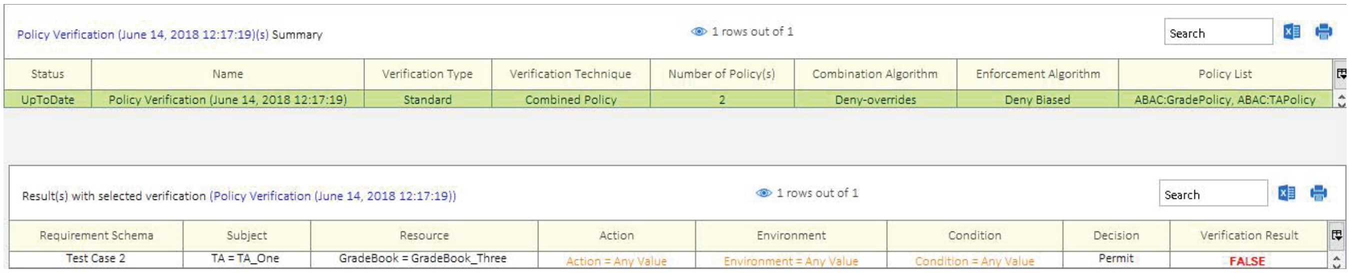
Fig. 20. Individual Security Requirement

10 Policy Verification/Analyzing Results - Test Case 2 (Not Protected Resource)

Now that we are ready to test our policies let's discuss the error we will be looking at in this second example. When policies are designed there is potential for a "Not Protected Resource" error being created by mistake. A Not Protected Resource error occurs when a resource is created but without protection from any rules.

For example, when the policy author was designing the logic for these university policies; the author created a resource "GradeBook_Three" with no protections. This means there are not currently any rules defined that are giving a decision for an access request to the resource. This Not Protected Resource error is not caused by any specific rules in either of our policies; it is caused due to a lack of rules created to cover this resource.

Next, we will run one "Combined Policy" Verification to reveal the Not Protected Resource error that is present in our policies. To do this, we will select Test Case 2 (security requirement) and GradePolicy & TAPolicy as a Combined Policy Verification and analyze our verification result. Again, this will have already been done for you if you open Project File: UniversityTestCase2.



Policy Verification (June 14, 2018 12:17:19)(s) Summary

1 rows out of 1

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:17:19)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

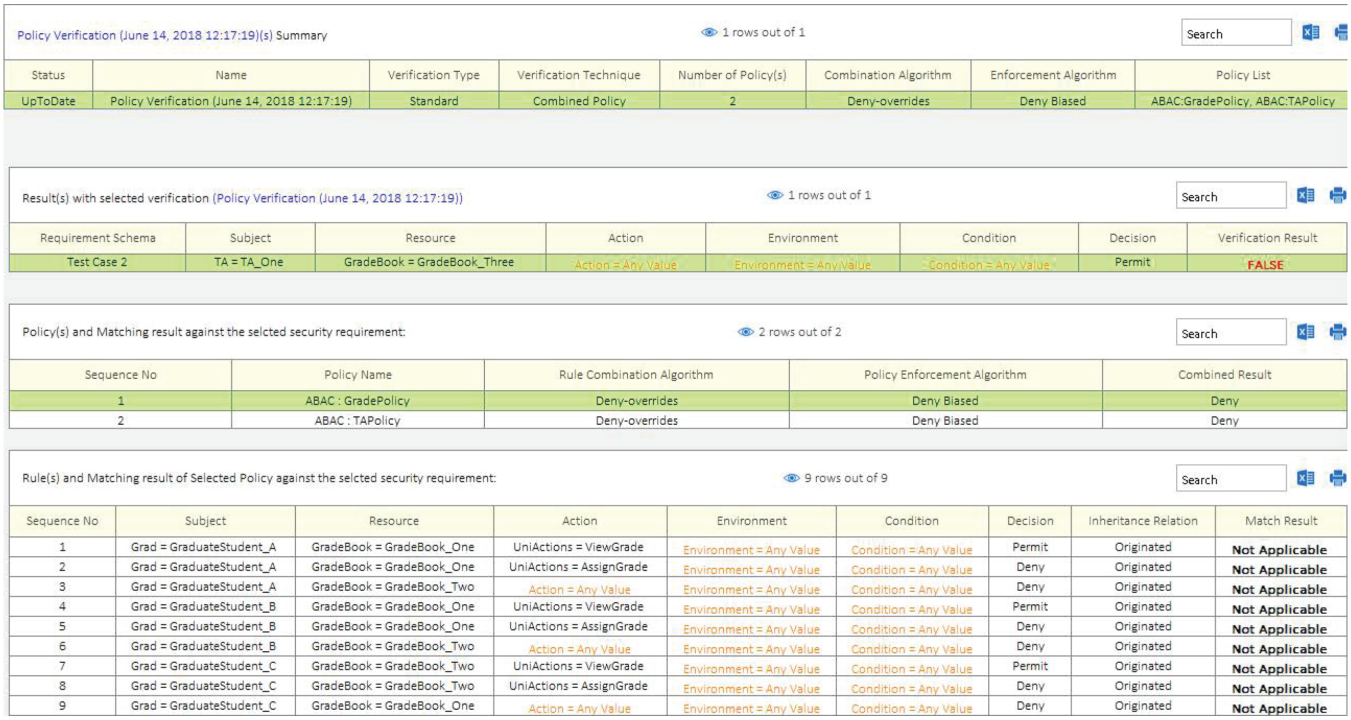
Result(s) with selected verification (Policy Verification (June 14, 2018 12:17:19))

1 rows out of 1

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 2	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	FALSE

Fig. 21. Combined Policy x Test Case 2

By clicking on the Verification Result, we can analyze deeper the reasoning for the "False" result we have received. Here is where we will notice we have not created any Rules that are attached to Resource = GradeBook_Three. We see this by noticing that every "Match Result" is "Not Applicable" whereas if there were Rules protecting this resource we would have seen at least one Rule with a (Permit or Deny) Match Result.



Policy Verification (June 14, 2018 12:17:19)(s) Summary

1 rows out of 1

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:17:19)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 12:17:19))

1 rows out of 1

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 2	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement:

2 rows out of 2

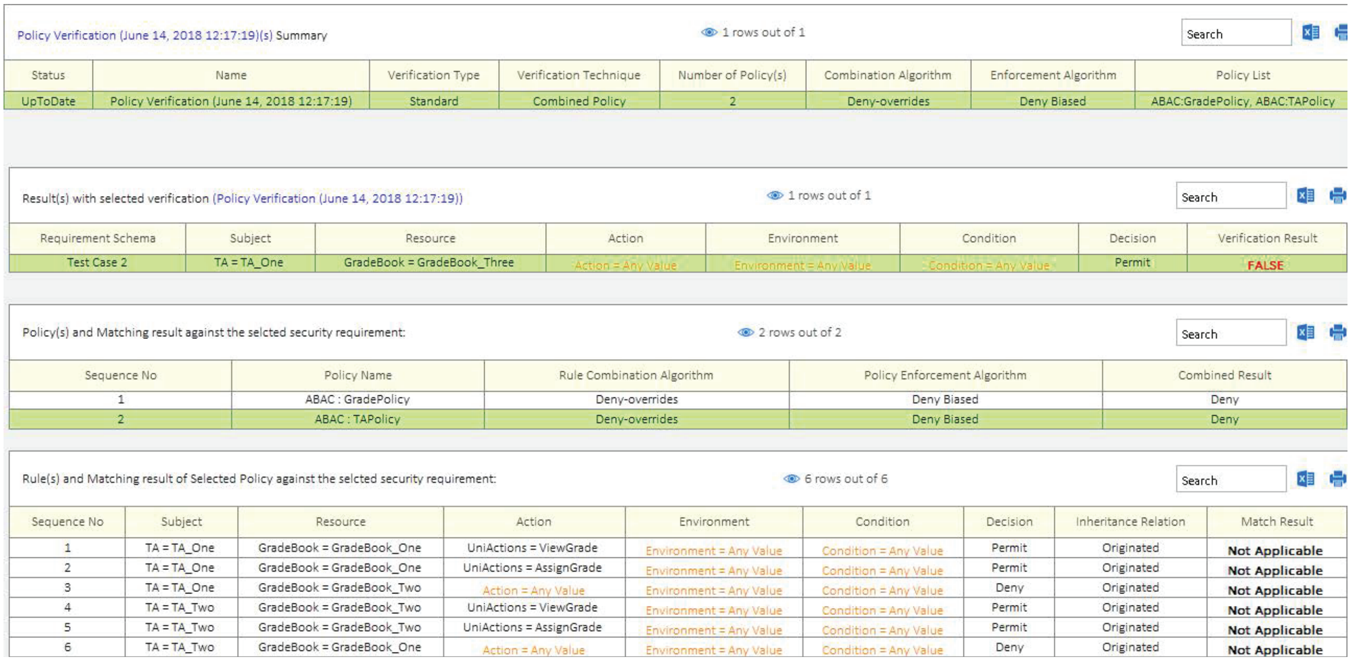
Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement:

9 rows out of 9

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
2	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
3	Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
4	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
5	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
6	Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
7	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
8	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
9	Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**

Fig. 22. GradePolicy: Match Results



Policy Verification (June 14, 2018 12:17:19)(s) Summary

1 rows out of 1

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:17:19)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 12:17:19))

1 rows out of 1

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 2	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement:

2 rows out of 2

Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement:

6 rows out of 6

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**

Fig. 23. TAPolicy: Match Results

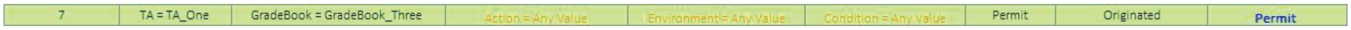
11 RESOLVING THIS ERROR - TEST CASE 2 (NOT PROTECTED RESOURCE)

To eliminate a Not Protected Resource vulnerability the policy author would need to define a specific rule for the unprotected resource (GradeBook_Three) and then test again to verify the intended access decision is being made based on this new rule design.

For example, if we're to add this rule below to the TAPolicy…

TAPolicy: Add (1) New Rule:

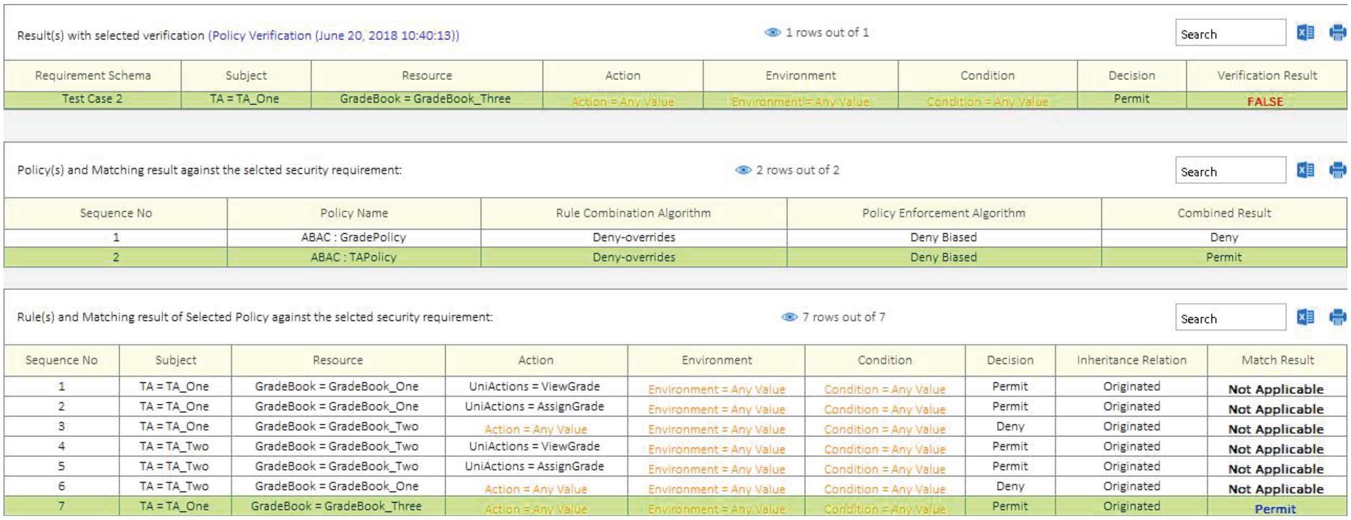
(Rule No. = 7) → (TA = TA_One) → (Action = Any Value) → (Resource = GradeBook_Three) → decision = Permit



7	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	Originated	Permit

Fig. 24. TAPolicy: New Rule (7)

Then retest using the same Policy Verification selections as last time we will get the same False Verification result due to our Algorithm selections. However, we can see in the Match Results that we have provided a rule for the system to evaluate for TA_One accessing this Resource.



Result(s) with selected verification (Policy Verification (June 20, 2018 10:40:13))

1 rows out of 1

Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 2	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement:

2 rows out of 2

Search

Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Permit

Rule(s) and Matching result of Selected Policy against the selcted security requirement:

7 rows out of 7

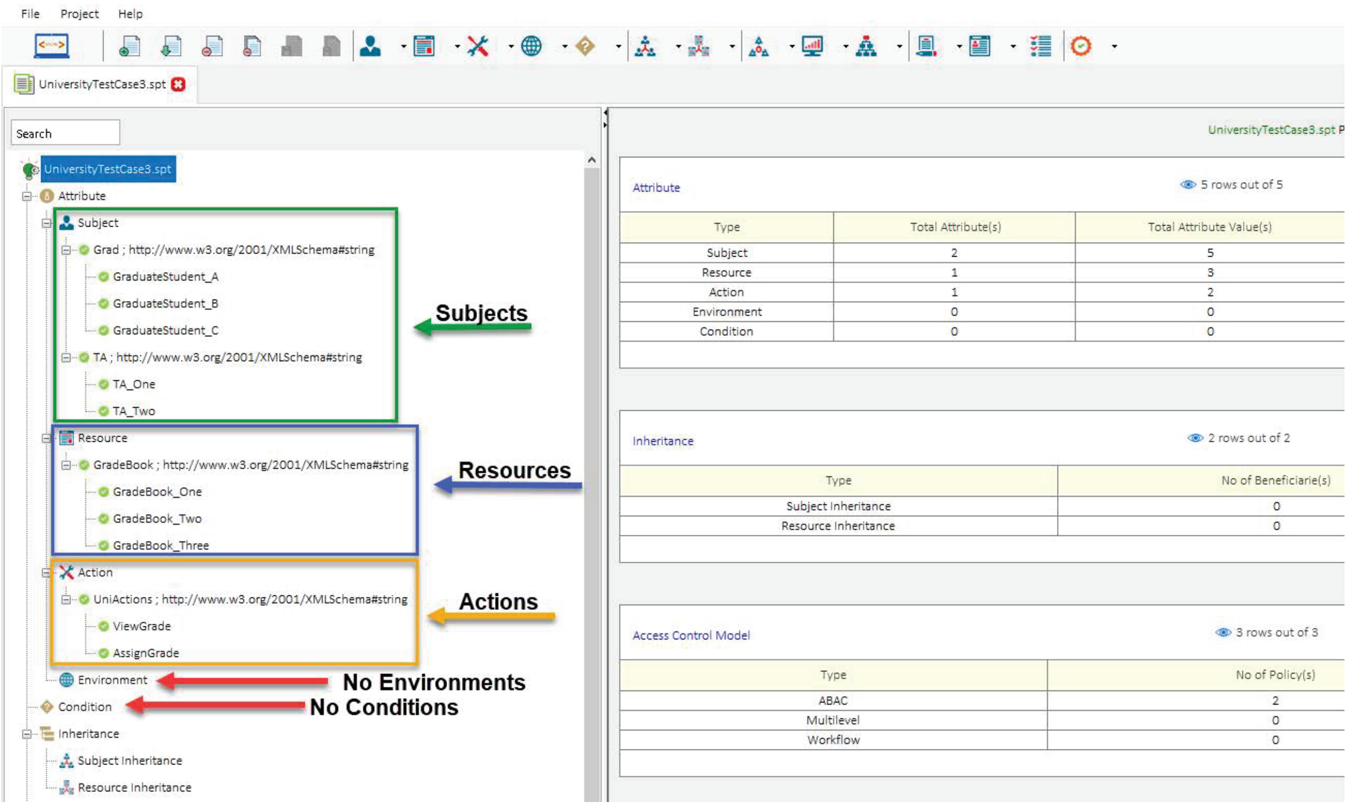
Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	Not Applicable
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	Not Applicable
7	TA = TA_One	GradeBook = GradeBook_Three	Action = Any Value	Environment = Any Value	Condition = Any Value	Permit	Originated	Permit

Fig. 25. Updated Policy: Resource Now Protected

12 Setting Up The Policies – Test Case 3 (Undecided Rule)

This university example contains two policies (GradePolicy & TAPolicy). The attributes in this example have not been changed from previous Test Case 2. The Attribute/Attribute Values included in these policies are as shown in Figure 26.



Attribute — 5 rows out of 5

Type	Total Attribute(s)	Total Attribute Value(s)
Subject	2	5
Resource	1	3
Action	1	2
Environment	0	0
Condition	0	0

Inheritance — 2 rows out of 2

Type	No of Beneficiarie(s)
Subject Inheritance	0
Resource Inheritance	0

Access Control Model — 3 rows out of 3

Type	No of Policy(s)
ABAC	2
Multilevel	0
Workflow	0

Fig. 26. Test Case 3

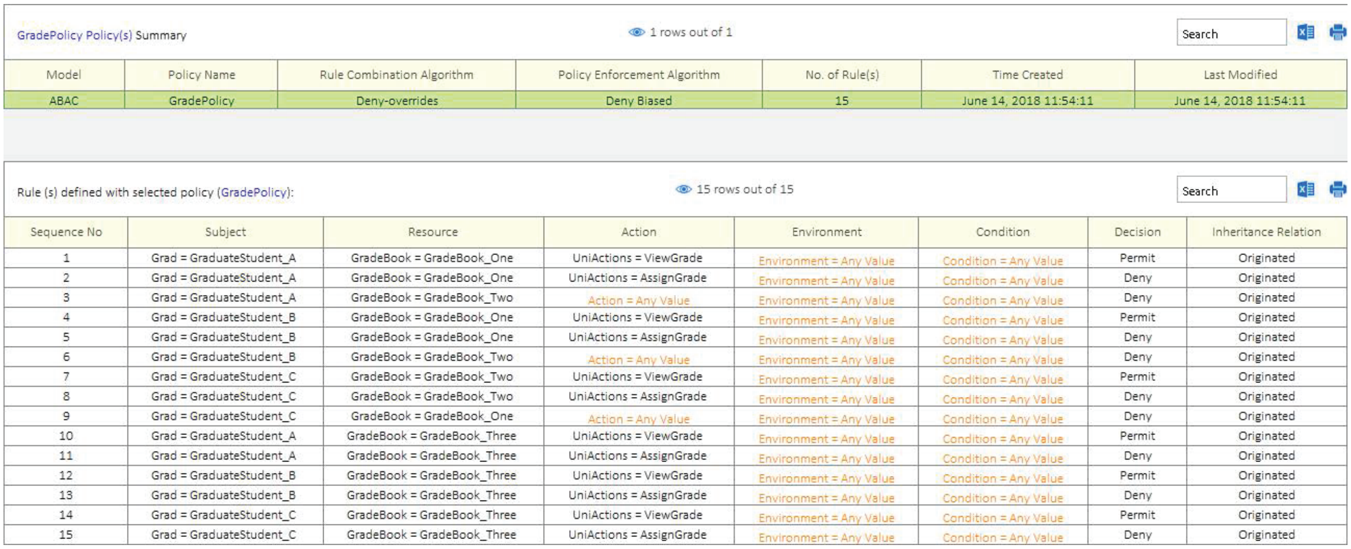
13 Modeling Your Policy – Test Case 3 (Undecided Rule)

Now that we have entered our attributes we can model our two policies (GradePolicy & TAPolicy). See the list below of the rules contained in each of these policies. You can open a "New (blank) Project" and build these policies by entering the following rules below:

GradePolicy:
(Grad = GraduateStudent_A, ViewGrade, GradeBook_One) → Permit
(Grad = GraduateStudent_A, AssignGrade, GradeBook_One) →Deny
(Grad = GraduateStudent_A, Any Action, GradeBook_Two) →Deny
(Grad = GraduateStudent_A, ViewGrade, GradeBook_Three) →Permit
(Grad = GraduateStudent_A, AssignGrade, GradeBook_Three) →Deny
(Grad = GraduateStudent_B, ViewGrade, GradeBook_One) →Permit
(Grad = GraduateStudent_B, AssignGrade, GradeBook_One) →Deny
(Grad = GraduateStudent_B, Any Action, GradeBook_Two) →Deny
(Grad = GraduateStudent_B, ViewGrade, GradeBook_Three) →Permit
(Grad = GraduateStudent_B, AssignGrade, GradeBook_Three) →Deny
(Grad = GraduateStudent_C, ViewGrade, GradeBook_Two) →Permit
(Grad = GraduateStudent_C, AssignGrade, GradeBook_Two) →Deny
(Grad = GraduateStudent_C, Any Action, GradeBook_One) →Deny
(Grad = GraduateStudent_C, ViewGrade, GradeBook_Three) →Permit
(Grad = GraduateStudent_C, AssignGrade, GradeBook_Three) →Deny

TAPolicy:
(TA = TA_One, ViewGrade, GradeBook_One) →Permit
(TA = TA_One, AssignGrade, GradeBook_One) →Permit
(TA = TA_One, Any Action, GradeBook_Two) →Deny
(TA = TA_One, ViewGrade, GradeBook_Three) →Permit
(TA = TA_One, AssignGrade, GradeBook_Three) →Permit
(TA = TA_Two, ViewGrade, GradeBook_Two) →Permit
(TA = TA_Two, AssignGrade, GradeBook_Two) →Permit
(TA = TA_Two, Any Action, GradeBook_One) →Deny

After entering the rules above your modeled policies should look like the screenshots below. If you did not create your own Project File, you can simply open Security Policy Tool – Project File: UniversityTestCase3 and these policies will have been already created for you.



GradePolicy Policy(s) Summary

1 rows out of 1

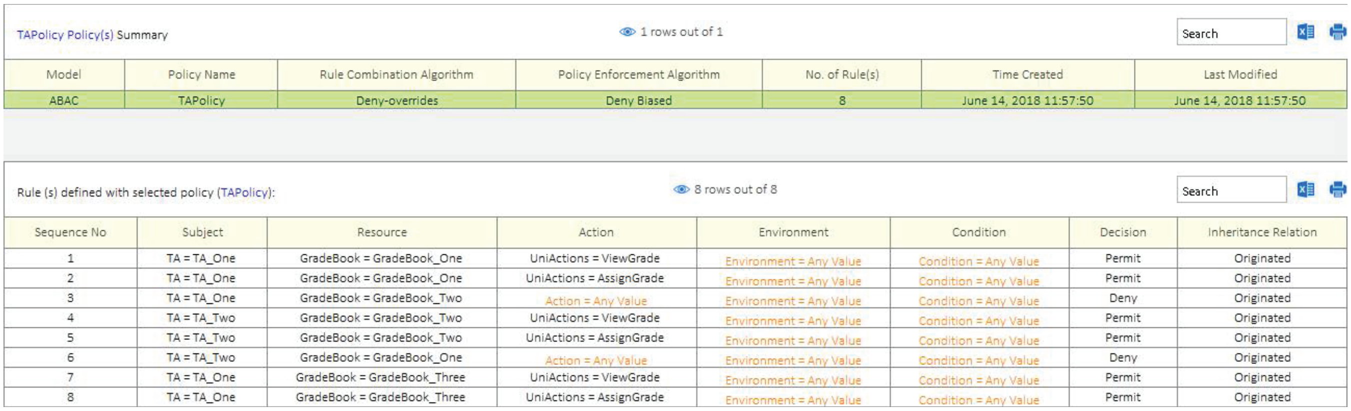
Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	GradePolicy	Deny-overrides	Deny Biased	15	June 14, 2018 11:54:11	June 14, 2018 11:54:11

Rule (s) defined with selected policy (GradePolicy):

15 rows out of 15

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
3	Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
6	Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
7	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
8	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
9	Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
10	Grad = GraduateStudent_A	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
11	Grad = GraduateStudent_A	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
12	Grad = GraduateStudent_B	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
13	Grad = GraduateStudent_B	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated
14	Grad = GraduateStudent_C	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
15	Grad = GraduateStudent_C	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated

Fig. 27. GradePolicy



TAPolicy Policy(s) Summary

1 rows out of 1

Model	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	No. of Rule(s)	Time Created	Last Modified
ABAC	TAPolicy	Deny-overrides	Deny Biased	8	June 14, 2018 11:57:50	June 14, 2018 11:57:50

Rule (s) defined with selected policy (TAPolicy):

8 rows out of 8

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated
7	TA = TA_One	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
8	TA = TA_One	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated

Fig. 28. TAPolicy

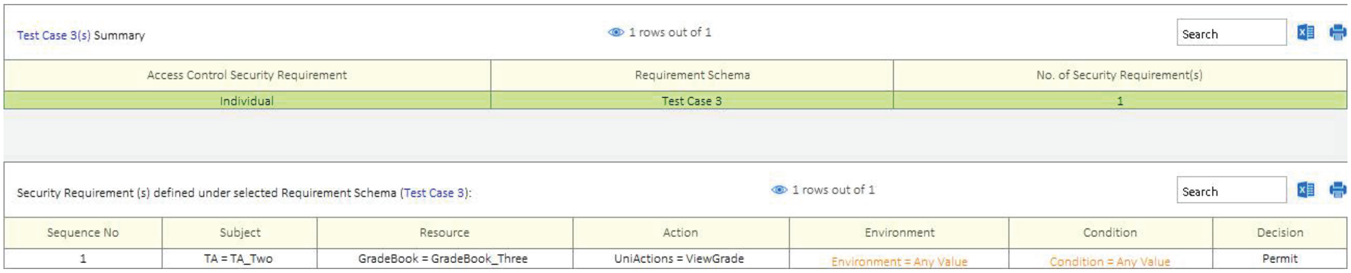
14 Individual Security Requirements - Test Case 3 (Undecided Rule)

The final step before analyzing these policies for errors is to create individual security requirements to use for testing. If you are building a "New (blank) Project" on your own you will enter the following security requirement below:

Individual Security Requirement:

(TA = TA_Two) & (Action = ViewGrade) & (GradeBook = GradeBook_Three) →decision = Permit

After entering the rule above your individual security requirement should look like the screenshot below. If you did not create your own Project File you can simply open Security Policy Tool – Project File: UniversityTestCase3 and this requirement will have been already created for you.



Test Case 3(s) Summary

1 rows out of 1

Search

Access Control Security Requirement	Requirement Schema	No. of Security Requirement(s)
Individual	Test Case 3	1

Security Requirement (s) defined under selected Requirement Schema (Test Case 3):

1 rows out of 1

Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision
1	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit

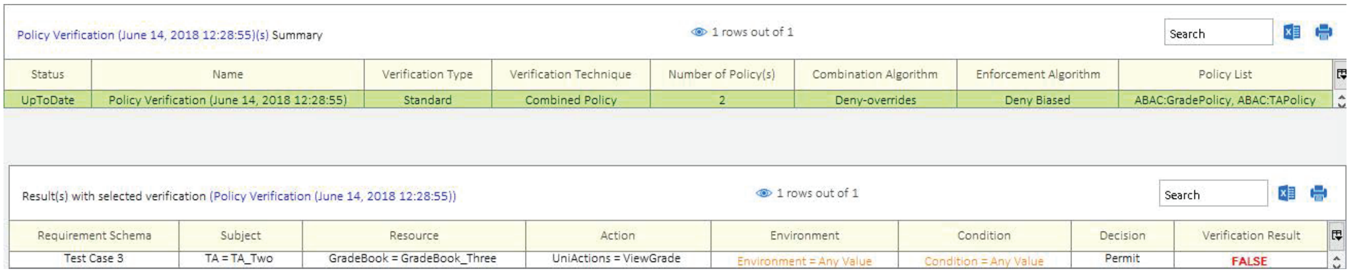
Fig. 29. Individual Security Requirement

15 Policy Verification/Analyzing Results - Test Case 3 (Undecided Rule)

Now that we are ready to test our policies let's discuss the error we will be looking at in this third example. When policies are designed there is potential for an "Undecided Rule" error being created. An Undecided Rule error occurs when your policy contains rules that are not entirely defined or are missing a step.

For example, when the policy author was designing the logic for these university policies; the author created rules for all Subjects to access "GradeBook_Three" but did not define access rules for TA = TA_Two. In this situation, if TA_Two were to attempt to take action on "GradeBook_Three," the system would be forced to make a default decision instead of a defined decision. This may create a security vulnerability due to your system's default evaluation decision being different than what you previously intended. Similar to the "Not Protected Resource" example previously, this error is caused due to the author missing rules. It is not caused due to flawed interpretation of existing rules contained in either of our policies as was the case in Test Case 1 (Rule Conflict).

Next, we will run one "Combined Policy" Verification to reveal the Undecided Rule error that is present in our policies. To do this, we will select Test Case 3 (security requirement) and GradePolicy & TAPolicy as a Combined Policy Verification and analyze our verification result. Again, this will have already been done for you if you open Project File: UniversityTestCase3.



Policy Verification (June 14, 2018 12:28:55)(s) Summary

1 rows out of 1

Search

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:28:55)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 12:28:55))

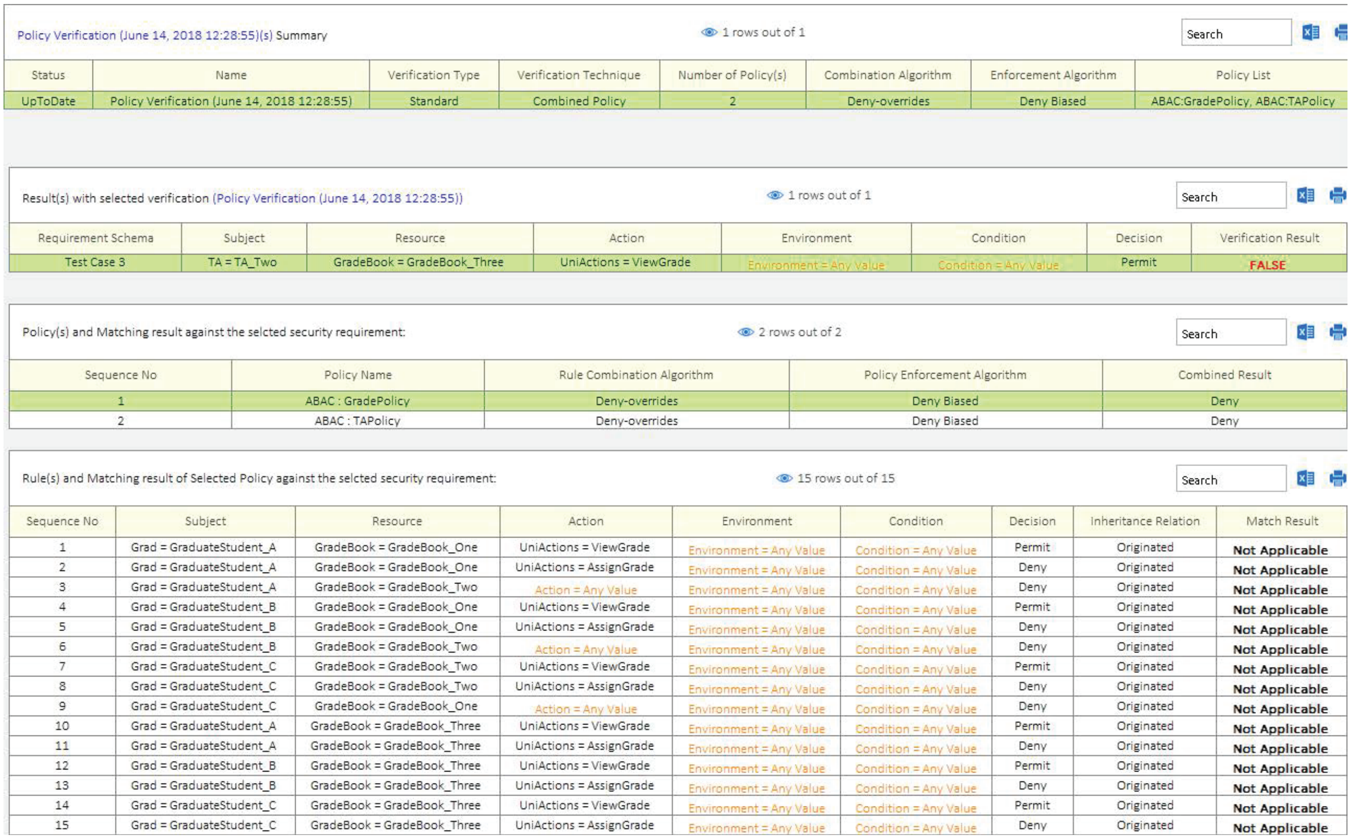
1 rows out of 1

Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 3	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

Fig. 30. Combined Policy x Test Case 3

Like we did in the "Not Protected Resource" example, by clicking on the Verification Result we can analyze deeper the reasoning for the "False" result we have received. Here is where we would notice we have not created any Rules that are attached to Subject = TA_Two taking action on Resource = GraduateBook_Three. We can see this by noticing that every "Match Result" is "Not Applicable" whereas if there were Rules existing for TA_Two and Resource = GraduateBook_Three we would have at least seen one Rule with a (Permit or Deny) Match Result.



Policy Verification (June 14, 2018 12:28:55)(s) Summary — 1 rows out of 1 — Search

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:28:55)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 12:28:55)) — 1 rows out of 1 — Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 3	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

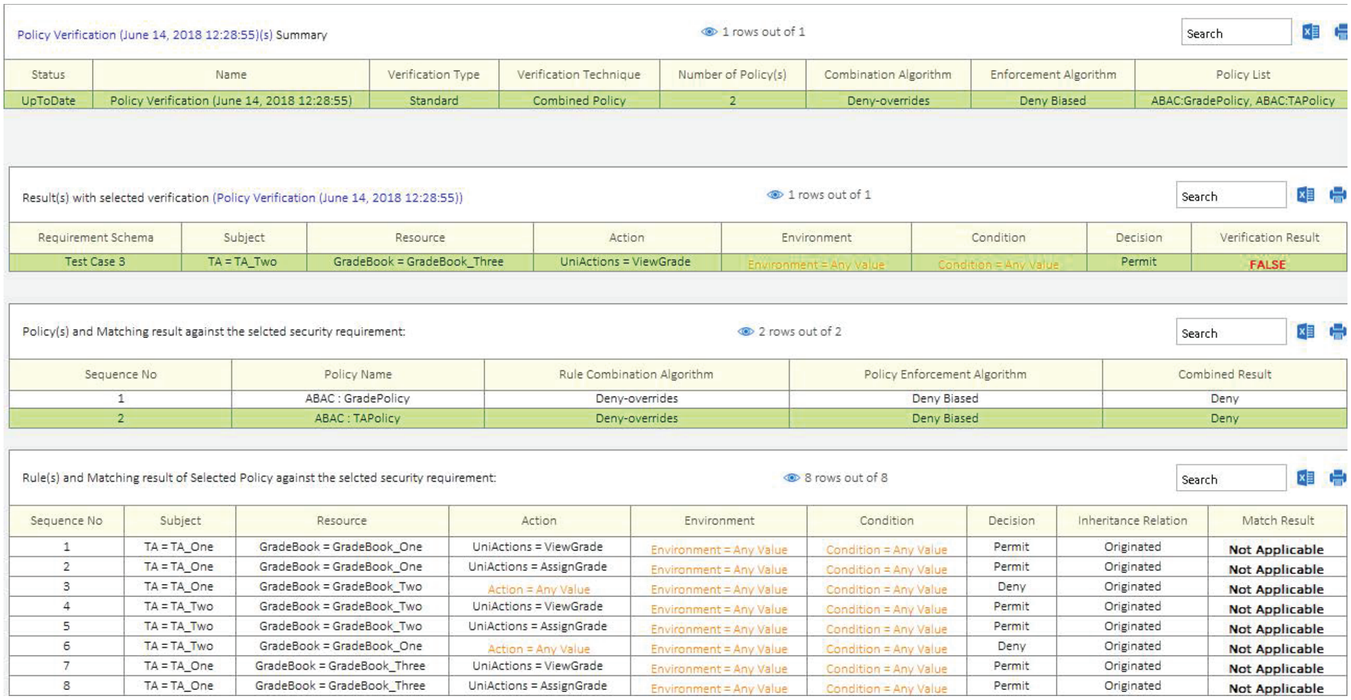
Policy(s) and Matching result against the selcted security requirement: — 2 rows out of 2 — Search

Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement: — 15 rows out of 15 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
2	Grad = GraduateStudent_A	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
3	Grad = GraduateStudent_A	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
4	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
5	Grad = GraduateStudent_B	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
6	Grad = GraduateStudent_B	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
7	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
8	Grad = GraduateStudent_C	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
9	Grad = GraduateStudent_C	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
10	Grad = GraduateStudent_A	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
11	Grad = GraduateStudent_A	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
12	Grad = GraduateStudent_B	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
13	Grad = GraduateStudent_B	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
14	Grad = GraduateStudent_C	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
15	Grad = GraduateStudent_C	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**

Fig. 31. GradePolicy: Match Results



Policy Verification (June 14, 2018 12:28:55)(s) Summary — 1 rows out of 1 — Search

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 14, 2018 12:28:55)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 14, 2018 12:28:55)) — 1 rows out of 1 — Search

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 3	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement: — 2 rows out of 2 — Search

Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Deny

Rule(s) and Matching result of Selected Policy against the selcted security requirement: — 8 rows out of 8 — Search

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
7	TA = TA_One	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
8	TA = TA_One	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**

Fig. 32. TAPolicy: Match Results

As you can see there has not been a rule defined for TA_Two → Action → GraduateBook_Three which is known as an Undecided Rule error.

16 Resolving This Error - Test Case 3 (Undecided Rule)

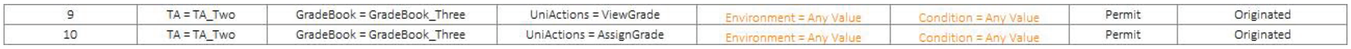
To solve this error, the policy author would need to define specific rules for all subject attributes (e.g., include TA_Two) in any policies that determine TA access requests to GraduateBook_Three.

For example, adding these rules below to the TAPolicy for our specific example…

TAPolicy: Add (2) New Rules:

(Rule No. = 9) → (TA = TA_Two) → (Action = ViewGrade) → (Resource = GradeBook_Three) → decision = Permit

(Rule No. = 10) → (TA = TA_Two) → (Action = AssignGrade) → (Resource = GradeBook_Three) → decision = Permit

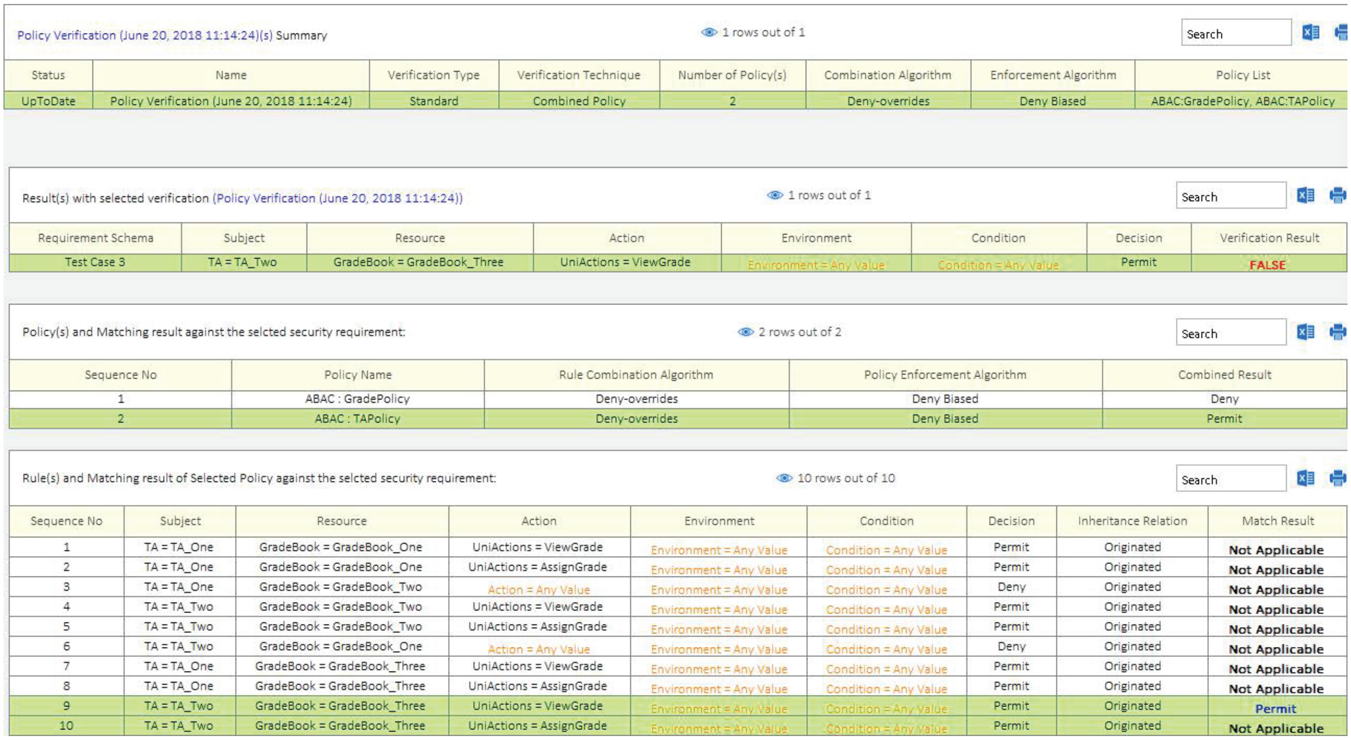


9	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated
10	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated

Fig. 33. TAPolicy: New Rules (9,10)

Now, looking out our Verification results and Match Results we will see that we no longer have an "Undecided Rule" error occurring. The Verification Result is still "False" due to our choices in our Combination Algorithm = Deny-overrides and Enforcement Algorithm = Deny Biased.

For example, GradePolicy has no rules related to the security requirement (TA_Two → View-Grades → GradeBook_Three) we are using for testing which is why see all Match Rules = Not Applicable. Due to our selection to use Deny Biased for our Enforcement Algorithm the "Combined Result" for GradePolicy = Deny. However, in the case of the TAPolicy we have the Combined Result = Permit due to the new rules we added (e.g., see new Rule 9 below). Hence, we have opposing Combined Results (GradePolicy = Deny; TAPolicy = Permit). Finally, the Combination Algorithm = Deny-overrides makes a definitive answer for our Verification Results. The Deny-overrides selection overrules the Permit result from the TAPolicy in favor of the Deny result from the GradePolicy to make the final Verification Result = False.



Policy Verification (June 20, 2018 11:14:24)(s) Summary

1 rows out of 1

Status	Name	Verification Type	Verification Technique	Number of Policy(s)	Combination Algorithm	Enforcement Algorithm	Policy List
UpToDate	Policy Verification (June 20, 2018 11:14:24)	Standard	Combined Policy	2	Deny-overrides	Deny Biased	ABAC:GradePolicy, ABAC:TAPolicy

Result(s) with selected verification (Policy Verification (June 20, 2018 11:14:24))

1 rows out of 1

Requirement Schema	Subject	Resource	Action	Environment	Condition	Decision	Verification Result
Test Case 3	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	FALSE

Policy(s) and Matching result against the selcted security requirement:

2 rows out of 2

Sequence No	Policy Name	Rule Combination Algorithm	Policy Enforcement Algorithm	Combined Result
1	ABAC : GradePolicy	Deny-overrides	Deny Biased	Deny
2	ABAC : TAPolicy	Deny-overrides	Deny Biased	Permit

Rule(s) and Matching result of Selected Policy against the selcted security requirement:

10 rows out of 10

Sequence No	Subject	Resource	Action	Environment	Condition	Decision	Inheritance Relation	Match Result
1	TA = TA_One	GradeBook = GradeBook_One	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
2	TA = TA_One	GradeBook = GradeBook_One	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
3	TA = TA_One	GradeBook = GradeBook_Two	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
4	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
5	TA = TA_Two	GradeBook = GradeBook_Two	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
6	TA = TA_Two	GradeBook = GradeBook_One	Action = Any Value	Environment = Any Value	Condition = Any Value	Deny	Originated	**Not Applicable**
7	TA = TA_One	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
8	TA = TA_One	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**
9	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = ViewGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Permit**
10	TA = TA_Two	GradeBook = GradeBook_Three	UniActions = AssignGrade	Environment = Any Value	Condition = Any Value	Permit	Originated	**Not Applicable**

Fig. 34. Updated Results: No Undecided Rule

17 Conclusion

Now you should have a better understanding of what to look for as you go onto verify your access control policies with Security Policy Tool. In addition to this document there are other resources located in the Learning Center in your My account page that will help you start leveraging Security Policy Tool to prevent access control leaks, today!

If you have not yet, download Security Policy Tool – Lite Version for FREE now! Close the door the Access Control Leaks and save time and cost creating, modeling, testing, and verifying your access control policies, today.

Click here to begin securing your policies now → Lite Version.

InfoBeyond Technology, LLC is an innovative company specializing in Network, Machine Learning and Data Security within the Information Technology industry. The mission of InfoBeyond is to research, develop, and deliver viable software products for network communication and security. Some of our research is sponsored by Department of Defense, Department of Energy, Missile Defense Agency, Department of Transportation, NIST (National Institute of Standards and Technology), etc. Security Policy Tool is Awarded the 2017 Innovative Security Solution Award at the 2017 Big Data and SDN/NFV Summit. NXdrive is a fragment-based cybersecurity storage system and more information can be found at www.NXdrive.com. The company is featured as one of 50 fast growth IT small businesses in 2017 by The Silicon Review.