**University Policy Test Case**

**1. Introduction**

The University policy is an ARBAC policy that includes rules for assignment of users to various student roles and employee roles [1, 2]. There are 32 roles and 449 rules in the original policy [3].

The test cases described in this document are derived from the University policy. Each test case consists of rules from the original University policy and rules or resources created by us (which are highlighted) to embed errors.

**2. Basic Test Case without Error**

**Roles:** Grad (GraduateStudent\_A, GraduateStudent\_B, GraduateStudent\_C)

TA (TA\_One, TA\_Two)

**Resource:** GradeBook (GradeBook\_One, GradeBook\_Two)

**Action:** ViewGrade, AssignGrade

**Rules:**

**GradPolicy:**

(GraduateStudent\_A, ViewGrade, GradeBook\_One) ->Permit

(GraduateStudent\_A, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_A, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_B, ViewGrade, GradeBook\_One)->Permit

(GraduateStudent\_B, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_B, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_C, ViewGrade, GradeBook\_Two) ->Permit

(GraduateStudent\_C, AssignGrade, GradeBook\_Two) ->Deny

(GraduateStudent\_C, Action: Any, GradeBook\_One) ->Deny

**TAPolicy:**

(TA\_One, ViewGrade, GradeBook\_One) ->Permit

(TA\_One, AssignGrade, GradeBook\_One)->Permit

(TA\_One, Action: Any, GradeBook\_Two) ->Deny

(TA\_Two, ViewGrade, GradeBook\_Two) ->Permit

(TA\_Two, AssignGrade, GradeBook\_Two) ->Permit

(TA\_Two, Action: Any, GradeBook\_One) ->Deny

**3. Test case with Error Type #1 <conflict>**

**Roles:** Grad (GraduateStudent\_A, GraduateStudent\_B, GraduateStudent\_C)

TA (GraduateStudent\_C (NOTE: this modification is used to generate type#1 error, causing conflict in GraduateStudent\_C’s access to GradeBook\_One and GradeBook\_Two), TA\_Two)

**Resource:** GradeBook (GradeBook\_One, GradeBook\_Two)

**Action:** ViewGrade, AssignGrade

**Rules:**

**GradPolicy:**

(GraduateStudent\_A, ViewGrade, GradeBook\_One) ->Permit

(GraduateStudent\_A, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_A, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_B, ViewGrade, GradeBook\_One)->Permit

(GraduateStudent\_B, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_B, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_C, ViewGrade, GradeBook\_Two) ->Permit

(GraduateStudent\_C, AssignGrade, GradeBook\_Two) ->Deny

(GraduateStudent\_C, Action: Any, GradeBook\_One) ->Deny

**TAPolicy:**

(GraduateStudent\_C, ViewGrade, GradeBook\_One) ->Permit

(GraduateStudent\_C, AssignGrade, GradeBook\_One) ->Permit (NOTE: this conflicts with Grad: GraduateStudent\_C, Action: Any, GradeBook\_One->Deny in GradPolicy)

(GraduateStudent\_C, Action: Any, GradeBook\_Two)->Deny (NOTE: This conflicts with Grad: GraduateStudent\_C, ViewGrade, GradeBook\_Two ->Permit in GradPolicy)

(TA\_Two, ViewGrade, GradeBook\_Two) ->Permit

(TA\_Two, AssignGrade, GradeBook\_Two) ->Permit

(TA\_Two, Action: Any, GradeBook\_One) ->Deny

**Property used to check the error:**

(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

**Result on ACPT (NuSMV Verification for Merged Policies):**

(Grad = GraduateStudent\_C & TA = GraduateStudent\_C) & (Action = viewGrade) & (GradeBook = GradeBook\_One) -> decision = Permit in GradPolicy **is False**

(Grad = GraduateStudent\_C & TA = GraduateStudent\_C) & (Action = viewGrade) & (GradeBook = GradeBook\_One) -> decision = Permit in TAPolicy **is True**

(Grad = GraduateStudent\_C & TA = GraduateStudent\_C) & (Action = viewGrade) & (GradeBook = GradeBook\_Two) -> decision = Permit in TAPolicy **is False**

(Grad = GraduateStudent\_C & TA = GraduateStudent\_C) & (Action = viewGrade) & (GradeBook = GradeBook\_Two) -> decision = Permit in GradPolicy **is True**

**4. Test case with Error Type #2 <no object>**

**Roles:** Grad (GraduateStudent\_A, GraduateStudent\_B, GraduateStudent\_C)

TA (TA\_One, TA\_Two)

**Resource:** GradeBook (GradeBook\_One, GradeBook\_Two, GradeBook\_Three (NOTE: this resource is added to generate type#2 error. It is not protected by any rule.))

**Action:** ViewGrade, AssignGrade

**Rules:**

**GradPolicy:**

(GraduateStudent\_A, ViewGrade, GradeBook\_One) ->Permit

(GraduateStudent\_A, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_A, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_B, ViewGrade, GradeBook\_One)->Permit

(GraduateStudent\_B, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_B, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_C, ViewGrade, GradeBook\_Two) ->Permit

(GraduateStudent\_C, AssignGrade, GradeBook\_Two) ->Deny

(GraduateStudent\_C, Action: Any, GradeBook\_One) ->Deny

**TAPolicy:**

(TA\_One, ViewGrade, GradeBook\_One) ->Permit

(TA\_One, AssignGrade, GradeBook\_One)->Permit

(TA\_One, Action: Any, GradeBook\_Two) ->Deny

(TA\_Two, ViewGrade, GradeBook\_Two) ->Permit

(TA\_Two, AssignGrade, GradeBook\_Two) ->Permit

(TA\_Two, Action: Any, GradeBook\_One) ->Deny

**Property used to check the error:**

(TA = TA\_One) & (Action = Any) & (GradeBook = GradeBook\_Three) -> decision = Permit

**Result on ACPT (NuSMV Verification for Combined Policies):**

(TA = TA\_One) & (Action = Any) & (GradeBook = GradeBook\_Three) -> decision = Permit **is False**

**5. Test Case with Error Type #3 <undecided>**

**Roles:** Grad (GraduateStudent\_A, GraduateStudent\_B, GraduateStudent\_C)

TA (TA\_One, TA\_Two)

**Resource:** GradeBook (GradeBook\_One, GradeBook\_Two, GradeBook\_Three (NOTE: this resource is added to generate type#3 error. Every role has been assigned access to it except TA\_Two.))

**Action:** ViewGrade, AssignGrade

**Rules:**

GradPolicy:

(GraduateStudent\_A, ViewGrade, GradeBook\_One) ->Permit

(GraduateStudent\_A, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_A, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_A, ViewGrade, GradeBook\_Three) ->Permit (NOTE: assign access for GraduateStudent\_A)

(GraduateStudent\_A, AssignGrade, GradeBook\_Three) ->Deny

(GraduateStudent\_B, ViewGrade, GradeBook\_One)->Permit

(GraduateStudent\_B, AssignGrade, GradeBook\_One) ->Deny

(GraduateStudent\_B, Action: Any, GradeBook\_Two) ->Deny

(GraduateStudent\_B, ViewGrade, GradeBook\_Three) ->Permit (NOTE: assign access for GraduateStudent\_A)

(GraduateStudent\_B, AssignGrade, GradeBook\_Three) ->Deny

(GraduateStudent\_C, ViewGrade, GradeBook\_Two) ->Permit

(GraduateStudent\_C, AssignGrade, GradeBook\_Two) ->Deny

(GraduateStudent\_C, Action: Any, GradeBook\_One) ->Deny

(GraduateStudent\_C, ViewGrade, GradeBook\_Three) ->Permit (NOTE: assign access for GraduateStudent\_C)

(GraduateStudent\_C, AssignGrade, GradeBook\_Three) ->Deny

TAPolicy:

(TA\_One, ViewGrade, GradeBook\_One) ->Permit

(TA\_One, AssignGrade, GradeBook\_One)->Permit

(TA\_One, Action: Any, GradeBook\_Two) ->Deny

(TA\_One, ViewGrade, GradeBook\_Three) ->Permit (NOTE: assign access for TA\_One)

(TA\_One, AssignGrade, GradeBook\_Three) ->Permit

(TA\_Two, ViewGrade, GradeBook\_Two) ->Permit

(TA\_Two, AssignGrade, GradeBook\_Two) ->Permit

(TA\_Two, Action: Any, GradeBook\_One) ->Deny

**Property used to check the error:**

(TA = TA\_Two) & (Action = viewGrade) & (GradeBook = GradeBook\_Three) -> decision = Permit

**Result on ACPT (NuSMV Verification for Combined Policies):**

(TA = TA\_Two) & (Action = viewGrade) & (GradeBook = GradeBook\_Three) -> decision = Permit **is False**

**References**

[1] Mikhail I. Gofman, Ruiqi Luo, Ayla C. Solomon, Yingbin Zhang, Ping Yang, and Scott D. Stoller. RBAC‐PAT: A Policy Analysis Tool for Role Based Access Control. Int'l Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2013.

[2] Scott D. Stoller, Ping Yang, C.R. Ramakrishnan, and Mikhail I. Gofman. Efficient Policy Analysis for Administrative Role Based Access Control. ACM CCS, 2007.

[3] http://www3.cs.stonybrook.edu/~stoller/ccs2007/university-policy.txt