**Continue Policy Test Cases**

**1. Introduction**

The Continue policy is used for a conference paper review system [1, 2]. Continue is a web-based software system that supports the submission, review, discussion, and notification phases of conferences, and it has been used by several conferences. There are 5 roles and more than 100 rules in the original policy [3].

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

**2. Basic Test Case without Error**

**Roles:** Reviewer, PCmember

**Users->Roles:** Reviewer (pcmember\_1, pcmember\_2, pcmember\_3)

**Action:** Read, MakeReview, Update

**ReviewStatus:** Submitted, Not\_Submitted

**Inheritance:** Beneficiary-> Reviewer (pcmember\_1, pcmember\_2, pcmember\_3), Inherited Values->PCmember

**Resource:** Paper (Paper\_1, Paper\_2, Paper\_3) Review (Review\_1, Review\_2, Review\_3)

**Rules (#12):**

**PCmember Policy:**

(pcmember, Read, Paper\_1)->Permit

(pcmember, Read, Paper\_2)->Permit

(pcmember, Read, Paper\_3) ->Permit

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_1) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_2) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_3) ->Deny

**Reviewer Policy:**

(pcmember\_1, ReviewStatus: Not\_Submitted, Update, Review\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, Update, Review\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Update, Review\_3) ->Permit

(pcmember\_1, ReviewStatus: Not\_Submitted, MakeReview, Paper\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, MakeReview, Paper\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, MakeReview, Paper\_3) ->Permit

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

**Roles**: Reviewer, PCmember

**Users->Roles**: Reviewer (pcmember\_1, pcmember\_2, pcmember\_3)

**Action**: Read, MakeReview, Update

**ReviewStatus**: Submitted, Not\_Submitted

**Inheritance**: Beneficiary-> Reviewer (pcmember\_1, pcmember\_2, pcmember\_3), Inherited Values->PCmember

**Resource**: Paper (Paper\_1, Paper\_2, Paper\_3) Review (Review\_1, Review\_2, Review\_3)

**Rules**:

**PCmember Policy**:

(pcmember, Read, Paper\_1)->Permit

(pcmember, Read, Paper\_2)->Permit

(pcmember, Read, Paper\_3) ->Permit

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_1) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_2) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_3) ->Deny

**Reviewer Policy**:

(pcmember\_1, ReviewStatus: Not\_Submitted, Update, Review\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, Update, Review\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Update, Review\_3) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Read, Review\_3) ->Permit (NOTE: This rule is added to generate type# 1 error. This rule allows pcmember\_3 to read Review\_3, which is in conflict with the last rule in PCmember Policy)

(pcmember\_1, ReviewStatus: Not\_Submitted, MakeReview, Paper\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, MakeReview, Paper\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, MakeReview, Paper\_3) ->Permit

**Property used to test the error**:

(PCmember = pcmember & Reviewer = pcmember\_3) & (ReviewStatus = Not\_Submitted) & (Access = Read) & (Review = Review\_3) -> decision = Permit

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

(PCmember = pcmember & Reviewer = pcmember\_3) & (ReviewStatus = Not\_Submitted) & (Access = Read) & (Review = Review\_3) -> decision = Permit **in PCmemberPolicy is False**

(PCmember = pcmember & Reviewer = pcmember\_3) & (ReviewStatus = Not\_Submitted) & (Access = Read) & (Review = Review\_3) -> decision = Permit **in ReviewerPolicy is True**

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

**Roles**: Reviewer, PCmember

**Users->Roles**: Reviewer (pcmember\_1, pcmember\_2, pcmember\_3)

**Action**: Read, MakeReview, Update

**ReviewStatus**: Submitted, Not\_Submitted

**Inheritance**: Beneficiary-> Reviewer (pcmember\_1, pcmember\_2, pcmember\_3), Inherited Values->PCmember

**Resource**: Paper (Paper\_1, Paper\_2, Paper\_3, Paper\_4 (NOTE: this resource is added to generate type# 2 error, because it is an object not protected by the access control rules)) Review (Review\_1, Review\_2, Review\_3)

**Rules**:

**PCmember Policy**:

(pcmember, Read, Paper\_1)->Permit

(pcmember, Read, Paper\_2)->Permit

(pcmember, Read, Paper\_3) ->Permit

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_1) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_2) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_3) ->Deny

**Reviewer Policy**:

(pcmember\_1, ReviewStatus: Not\_Submitted, Update, Review\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, Update, Review\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Update, Review\_3) ->Permit

(pcmember\_1, ReviewStatus: Not\_Submitted, MakeReview, Paper\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, MakeReview, Paper\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, MakeReview, Paper\_3) ->Permit

**Property used to test the error**:

(Reviewer = pcmember\_3) & (Access = MakeReview) & (Paper = Paper\_4 ) & -> decision = Permit

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

(Reviewer = pcmember\_3) & (Access = MakeReview) & (Paper = Paper\_4 ) & -> decision = Permit **is False**

**5. Test case with Error Type #3 (undecided)**

**Roles:** Reviewer, PCmember

**Users->Roles:** Reviewer (pcmember\_1, pcmember\_2, pcmember\_3)

**Action:** Read, MakeReview, Update

**ReviewStatus:** Submitted, Not\_Submitted

**Inheritance:** Beneficiary-> Reviewer (pcmember\_1, pcmember\_2, pcmember\_3), Inherited Values->PCmember

**Resource:** Paper (Paper\_1, Paper\_2, Paper\_3) Review (Review\_1, Review\_2, Review\_3, Review\_4)

**PCmember Policy:**

(pcmember, Read, Paper\_1)->Permit

(pcmember, Read, Paper\_2)->Permit

(pcmember, Read, Paper\_3) ->Permit

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_1) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_2) ->Deny

(pcmember, ReviewStatus: Not\_Submitted, Read, Review\_3) ->Deny

**Reviewer Policy:**

(pcmember\_1, ReviewStatus: Not\_Submitted, Update, Review\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, Update, Review\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Update, Review\_3) ->Permit

(pcmember\_1, ReviewStatus: Not\_Submitted, MakeReview, Paper\_1) ->Permit

(pcmember\_2, ReviewStatus: Not\_Submitted, MakeReview, Paper\_2) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, MakeReview, Paper\_3) ->Permit

(pcmember\_3, ReviewStatus: Not\_Submitted, Update, Review\_4) ->Permit (NOTE: this rule is added to generate type# 3 error. It only allows pcmember\_3 to update Review\_4, but it doesn’t assign any access for pcmember\_1 and pcmember\_2 to Review\_4)

**Property (used to test the error)**:

(Reviewer = pcmember\_2) & (ReviewStatus = Not\_Submitted) &(Access = Update) & (Review = Review\_4) -> decision = Permit

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

(Reviewer = pcmember\_2) & (ReviewStatus = Not\_Submitted) &(Access = Update) & (Review = Review\_4) -> decision = Permit **is False**

**References**

[1] Nan Zhang, Dimitar P. Guelev and Mark Ryan. Synthesising Verified Access Control Systems through Model Checking. Journal of Computer Security 16(1):1-61. 2007.

[2] S. Krishnamurthi. The Continue server. In Symposium on the Practical Aspects of Declarative Languages, January 2003.

[3] http://www.margrave-tool.org/v1+v2/margrave/versions/01-01/examples/continue/