Formal Specifications of Security Policy Models

Wolfgang Thumser
T-Systems GEI GmbH
Overview and Motivation

Structure of talk

– Position of FSPM within CC 3.1 and CEM 3.1
  • CC Requirements
  • CEM Requirements
  • National Scheme
– Realization of FSPM in terms of
  • Features and Properties
  • Security Functionality
  • Security Functional Requirements
– Formal Systems
  • Formal Proof of Security and Consistency
  • Proof of Concept (An FSPM Example)
– Summary
Position of Formal Security Policy Model (CC 3.1)

Form of Presentation
- Informal TSP-Model
- Semiformal TSP-Model
- ADV_SPM.1 Formal TSP-Model

Modeled Policies
- Principles (Rules)
- Characteristics

Properties
- ADV_SPM.1.1C
- ADV_SPM.1.2C
- ADV_SPM.1.3,4,5C

Features
- ADV_FSP.1.4C

Content of Presentation
- TSP Model
- ST
- TSF
- SFR
- FSP

Relationships among ADV constructs and SPM
Position of Formal Security Policy Model (CEM 3.1)

Section 11.7 (ADV_SPM) of CEM 3.1:

- Evaluation of sub-activity (ADV_SPM.1, section 11.7.1)
  - “There is no general guidance; the scheme should be consulted on this sub-activity”

- The national scheme of Germany by BSI provides
  - AIS 34: Evaluation Methodology for CC Assurance Classes for EAL5+
  - Effective for CC Version 2.1 and CEM Version 1.0
  - Adaption necessary for CEM Version 3.1
  - Ideas based on...
Position of Formal Security Policy Model (Scheme)

The following Interpretation by BSI is effective (Germany):

- Provides terminology and guidance
- Relates to CC Version 2.1 and CEM Version 1.0
- Needs adaption to CC 3.1 and CEM 3.1
- Proved useful in former evaluations of FSPMs
Realization of FSPM (Features & Properties)

Syntax (formal) and Semantics (informal) classify

– Features and Properties as the formal counterpart of
– Characteristics and Rules, which constitute the SPM
– The terms are related by interpretation
– To show that
  • The Characteristics enforce the Principles (Rules)
    one transforms the terms into their formal counterpart
    and formally proves that
  • The Features imply the Properties
– We achieve
  • Rigor, Precision and Consistency by formal treatment
Realization of FSPM (Example)

The Characteristic that

- only the administrator may modify the access rights of an object

is interpreting the following Feature

\[ \forall x \in \text{Sub} \forall y \in \text{Obj} : acc(\text{mod}(x, y)) \neq acc(y) \rightarrow \text{admi}(x) \]

where other axioms determine the behavior of predicate \text{admi}(), operation \text{mod}() and function \text{acc}() in

First order Predicate Logic
Security Functionality

According to definition of section 4 of part 1 in CC 3.1:

The TSF as a subset of the TOE ensures

- correct enforcement of the SFRs

The SFP (Security Function Policy) is

- expressible as a set of SFRs

Which elements of TSF support which SFRs?

- Question can be answered by means of a table
### Security Functionality

<table>
<thead>
<tr>
<th>Defined in</th>
<th>TSF element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFRs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PP</td>
<td>FDP_IFC.1</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FDP_ITT.1</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FMT_LIM.1</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FRU_FLT.2</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>FPT_ACC.1</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FDP_ACF.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FMT_SMF.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table: Relation of SFRs and SFs to the formal model
Security Functional Requirements

SFRs are being mapped to the respective

- Security Policy Description consisting of
  - Characteristics and Rules (Principles)

along with their formal counterparts

- Security Invariant Description determined by
  - Features and Properties

by means of the following table
## Security Functional Requirements

<table>
<thead>
<tr>
<th>TSF element</th>
<th>SFR</th>
<th>Security Policy Description (informal in section #)</th>
<th>Security Invariant Description (formal in section #)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Characteristic</td>
<td>Principle</td>
</tr>
<tr>
<td>SF1</td>
<td>FRU_FLT.2.1</td>
<td>char_1 in sec1</td>
<td>prin_1 in sec1</td>
</tr>
<tr>
<td>SF2</td>
<td>FMT_LIM.1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF4</td>
<td>FDP_IFC.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FDP_ITT.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF6</td>
<td>FDP_ACC.1.1</td>
<td>char_e in section 3</td>
<td>prin_e in section 3</td>
</tr>
<tr>
<td>SF8</td>
<td>FDP_ACF.1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FDP_ACF.1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF9</td>
<td>FMT_SMF.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF10</td>
<td>FRU_FLT.2.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Relation of SFRs and SFs to the formal model
Security Functional Requirements

The developer should have to argue if

- she abstains from formally modeling certain SFRs
- some of the SFRs are not covered by the model

According to state of the art

- IFC can always be modeled (if addressed in ST)
- strong arguments needed in case of abstention from modeling ACC

as outlined in former CC version 2.1
Formal Systems

According to CC 3.1, part 3, section A.5 Development:

– A Formal Specification consists of
  • Formal System based upon
  • well-established Mathematical Concepts
    – well-defined Semantics
    – Syntax

– Formal System
  • Formal Language over some finite Alphabet
  • Logical and Non-logical Axioms
  • Rules of Inference to construct
    – Formal Derivations of
    – Theorems
  • can be combinatorially manipulated and controlled
Formal Systems

Realizations of Formal Systems include

- First Order Predicate Logic
- Intuitionistic Logic
- Modal Logic
- Temporal Logic of Actions

Verification tools for Formal Systems include

- Isabelle
- MetaMath
- B-Method
- VSE II
- Autofocus
ADV_SPM.1.2C of CC 3.1 requires:

“For all policies that are modelled, the model shall define security for the TOE and provide a formal proof that the TOE cannot reach a state that is not secure.”

In terms of Formal Systems this translates to:

$$feat\_e := \{Ax1, Ax2, ..., AxN\} | \neg prop\_e =: thm\_e ,$$

where “$|$” denotes the derivation operator and $thm\_e$ formalizes secure state maintenance.
Formal Systems (Consistency)

Regarding Consistency §268 of CC 3.1 states:

“The confidence in the model is accompanied by a proof that it contains no inconsistencies.”

To achieve Consistency in spite of the incompleteness phenomenon:

– Conservative extensions by definition of and
– Interpretations (Models) of theories within consistent theories are consistent.

So consistency can be obtained by literature reference.
Proof of Concept (FSPM Example)

As an easy example of access control (FDP_ACC.1.1):
Emergency supply consisting of seven power engines

Characteristics:
- Subjects: Power switches changing power state of adjacent power engines
- Objects: Power engines
- Operatns: Change power state of adjacent engines
- Initially: All engines turned on

Principles:
- Cannot run into insecure state (all engines turned off)
Proof of Concept (FSPM Example)

To formalize the Security Policy we take some first order theory SET equipped with:

– logical and nonlogical axioms for finite sets.
– classical rules of inference (modus ponens, tnt, etc.)

for SET we may take e.g.

– ZF with Axiom of Infinity replaced by its negation
– equivalent to first order Peano Arithmetics (PA)
– well known to be consistent:
  • $\text{PRA} + \text{Ind}(\epsilon_0) \models \neg \text{Cons}(\text{SET})$
– SET proves induction for first order formulas
– is easy to operate with (e.g. formalizable in Isabelle)
Proof of Concept (FSPM Example)

Consider the following extension by definition of theory SET formalizing the characteristics in terms of features feat_e:

- Ax1: \( S = \{s| s : \{1,2,\ldots,7\} \rightarrow \{0,1\}\} \) (state definition from objects)

- Ax2: \( s_0 \in S \land \forall k \in \{1,2,\ldots,7\}: s_0(k) = 0 \land s_1 \in S \land \forall k \in \{1,2,\ldots,7\}: s_1(k) = 1 \) (insecure and secure state)

- Ax3: \( \forall X: \text{Sec}(X) \leftrightarrow \neg s_0 \in X \) (secure set)

- Ax4: \( \forall s \in S \forall i \in \{1,2,\ldots,6\} \forall k \in \{1,2,\ldots,7\}: (k=i \lor k=i+1 \rightarrow \text{op}(i,s)(k)=1 - s(k)) \land (\neg k=i \land \neg k=i+1 \rightarrow \text{op}(i,s)(k) = s(k)) \) (state operation of subjects)

- Ax5: \( A_0 = \{s_1\} \land \forall n A_{n+1} = A_n \cup \bigcup_{i \in \{1,2,\ldots,6\}} \text{op}(i,A_n) \land A = \bigcup_{i \in \mathbb{N}} A_i \) (achievable states)
Proof of Concept (FSPM Example)

Proof of Security

- Let prop_e consist of Thm_e:
  - Thm_e: Sec(A)
- Prove that
  - \{Ax1,\ldots,Ax5\} |- Sec(A) in SET
- (Sketch of proof) Consider
  - Def1: \(\forall s \in S: \text{odd}(s) \iff \exists n \in \mathbb{N}: s(1) + \ldots + s(7) = 2n+1\)
  - Def2: \(\forall X \subseteq S: \text{Odd}(X) \iff \forall s \in X: \text{odd}(s)\)
  - We prove by induction on \(n\) using Ind:
    - feat_e |- \(\forall n \in \mathbb{N}: \text{Odd}(A_n)\) yielding
      - feat_e |- Odd(A) (by definition of A) (**)
      - feat_e |- \(\forall X \subseteq S: \text{Odd}(X) \rightarrow \text{Sec}(X)\), since feat_e |- \(\neg \text{odd}(s_0)\) (***)
    - feat_e |- Odd(A) \rightarrow Sec(A) (by Subst. from (**)) (***)
- feat_e |- Sec(A) (by modus ponens from (**),(***)}, q.e.d.
Proof of Concept (FSPM Example)

Proof of Consistency

- SET + feat_e is consistent since
  - Ax1 to Ax5 are extensions by definition of SET and
  - SET is consistent in the first place

- Conservation of consistency
  - for most well established mathematical theories, which are known to be consistent

- Fulfillment of requirement §268 of CC 3.1
Summary

Impact of CC 3.1 requirements on formal security policy models

- Contribution to Security Functionality and Requirements
- Formalize Security Characteristics and Principles
- Formally prove properties of SPM
- Proof of Consistency of FSPM
- Developer has to show evidence

Proof of concept by means of Example

Topics not covered yet
- Formal Demonstration of Correspondence SPM <-> FSP