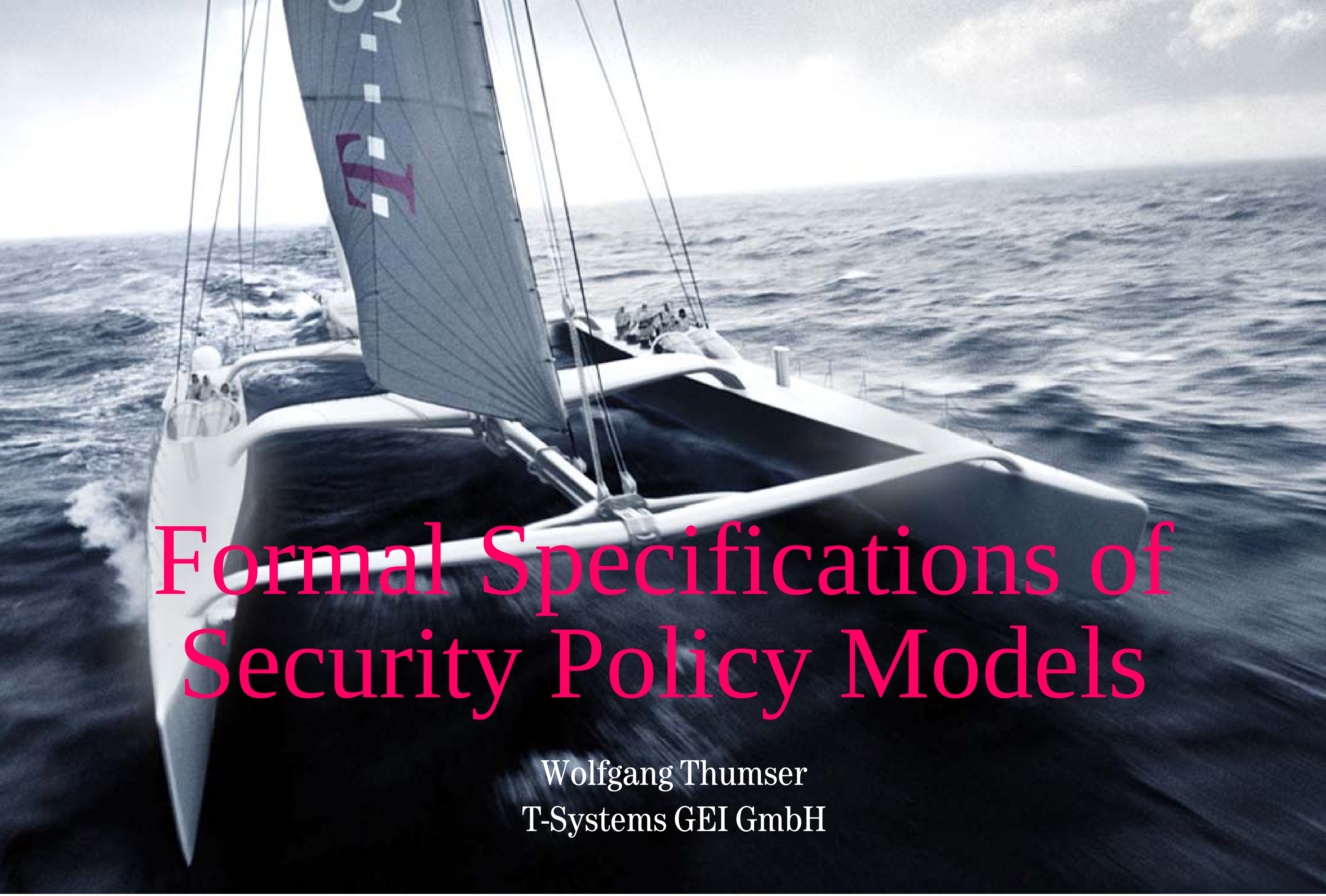


Formal Specifications of Security Policy Models

Wolfgang Thumser
T-Systems GEI GmbH

T · · · Systems ·

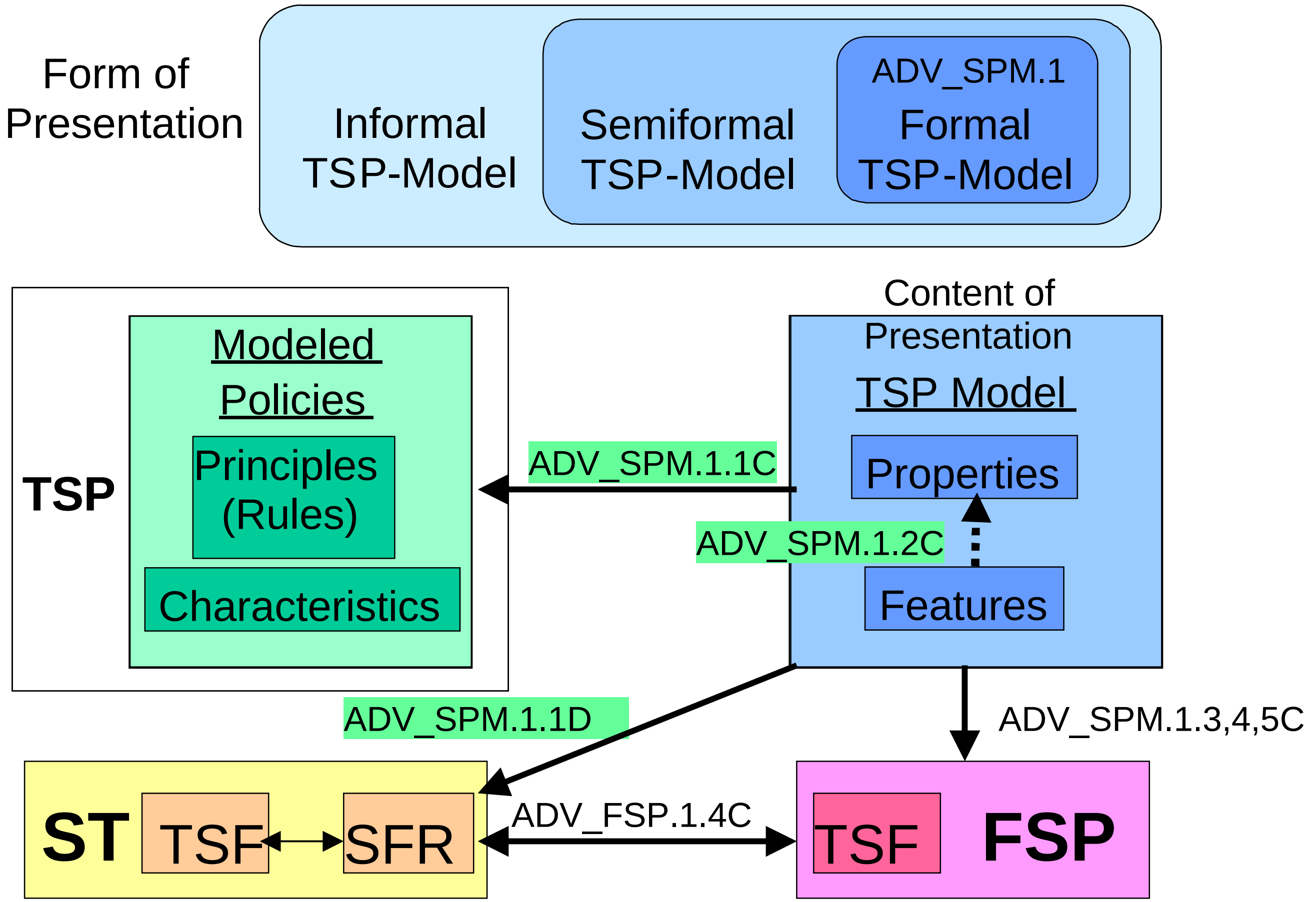
The 7th International Common Criteria Conference
19-21 September, Lanzarote, Spain

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)



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):

- AIS 39: Guideline for the Development and Evaluation of formal security policy models in the scope of ITSEC and Common Criteria, Version 1.1
- 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

– $\forall x \in Sub \forall y \in Obj : acc(\mathrm{mod}(x, y)) \neq acc(y) \rightarrow admi(x)$

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

Defined in	TSF element SFRs	1	2	3	4	5	6	7	8	9	10
PP	FDP_IFC.1				X						
	FDP_ITT.1				X						
	FMT_LIM.1		X								
	FRU_FLT.2	X									X
ST	FPT_ACC.1						X				
	FDP_ACF.1								X		
	FMT_SMF.1									X	

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

TSF element	SFR	Security Policy Description (informal in section #)		Security Invariant Description (formal in section #)	
		Characteristic	Principle	Feature	Property
SF1	FRU_FLT.2.1	char_1 in sec1	prin_1 in sec1	feat_e in sec1	prop_e in sec1
SF2	FMT_LIM.1.1			n/a	n/a
SF4	FDP_IFC.1				
	FDP_ITT.1				
SF6	FDP_ACC.1.1	char_e in section 3	prin_e in section 3	feat_e in section 3	prop_e in section 3
SF8	FDP_ACF.1.1				
	FDP_ACF.1.2				
SF9	FMT_SMF.1				
SF10	FRU_FLT.2.1				

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

Formal Systems (Formal Proof of Security)

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:

$$feat\_e := \{Ax1, Ax2, \ldots, AxN\} \mid\!- \; prop\_e =: thm\_e \,,$$

where "$\mid\!-$" 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:

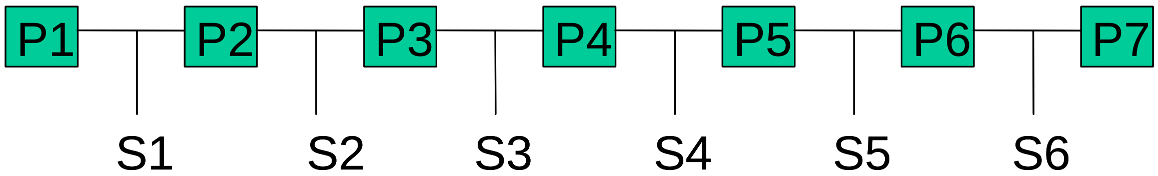
- Conservative extensions by definition of and
- Interpretations (Models) of theories within

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:
  - $PRA + Ind(\varepsilon_0) \mid\!- Cons(SET)$
- 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: $S = \{s \mid s : \{1,2,\ldots,7\} \to \{0,1\}\}$
  (state definition from objects)
– Ax2: $s_0 \, \varepsilon \, S \wedge \forall k \, \varepsilon \, \{1,2,\ldots,7\}: s_0(k) = 0 \wedge$
  $s_1 \, \varepsilon \, S \wedge \forall k \, \varepsilon \, \{1,2,\ldots,7\}: s_1(k) = 1$
  (insecure and secure state)
– Ax3: $\forall X: Sec(X) \leftrightarrow \neg \, s_0 \, \varepsilon \, X$
  (secure set)
– Ax4: $\forall s \, \varepsilon \, S \; \forall i \, \varepsilon \, \{1,2,\ldots,6\} \; \forall k \, \varepsilon \, \{1,2,\ldots,7\}$:
  $(k{=}i \vee k{=}i{+}1 \rightarrow op(i,s)(k) = 1 - s(k)) \wedge$
  $(\neg k{=}i \wedge \neg k{=}i{+}1 \rightarrow op(i,s)(k) = s(k))$
  (state operation of subjects)
– Ax5: $A_0 = \{s_1\} \wedge \forall n \; A_{n+1} = A_n \cup \bigcup_{i \, \varepsilon \, \{1,2,\ldots,6\}} op(i,A_n) \wedge$
  $A = \bigcup_{i \, \varepsilon \, \mathbb{N}} A_i$ (achievable states)

Proof of Concept (FSPM Example)

Proof of Security

- Let prop_e consist of Thm_e:
  - Thm_e: Sec(A)
- Prove that
  - {Ax1,…,Ax5} |- Sec(A) in SET
- (Sketch of proof) Consider
  - Def1: $\forall s \in S$: odd(s) $\leftrightarrow \exists n \in \mathbb{N}$: s(1) +…+ s(7) = 2n+1
  - Def2: $\forall X \subseteq S$: Odd(X) $\leftrightarrow \forall s \in X$: odd(s)
  - We prove by induction on n using Ind:
    - feat_e |- $\forall n \in \mathbb{N}$: Odd(A_n) yielding
  - feat_e |- Odd(A) (by definition of A) (*)
  - feat_e |- $\forall X \subseteq S$: Odd(X) $\rightarrow$ Sec(X), (**)
    since feat_e |- $\neg$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