Developing a Formal Security Policy Model for a Smart Card EAL6 Evaluation

Gerd Beuster (formerly T-Systems)  gb@fh-wedel.de
Karin Greimel (NXP)  karin.greimel@nxp.com
Motivation – Why EAL6?

- High Assurance
  - We want to give our customers a higher assurance that our new security IC satisfies the claimed security functional requirements.

- Documentation
  - Security Policy Model helps to have precise, clean, and consistent documentation.

- Security
  - Have an additional look from another perspective at the security functionality.
Overview

- Introduction to Formal Methods
  - Model Checking

- Common Criteria Certification EAL6 – Security Policy Model
  - What does it prove?
  - How do we implement it?
  - Example

- Conclusions
Formal Methods

Def.: Includes all mathematical techniques to specify and verify security and/or correctness of software or hardware.

- Formally specifying a system gives better understanding:
  - Forced to think about the details at the specification phase.
  - Forced to be precise at the specification phase.
  - No ambiguities, gives a common understanding of the TOE for architects, testers, developers ...

- Verification:
  - Gives a higher assurance of security and correctness.
  - Techniques:
    - refinement
    - theorem proving (natural deduction, math. induction -> proofs over infinite state space)
    - model checking, equivalence checking ...
Model Checking

- **Specification** describes the behavior of the hardware in terms of inputs and outputs.
  - For example as a temporal logic formula:
    \[ \text{always}((i=1) \rightarrow \text{next}(o=1)) \]
    'Every input \( i=1 \) must be followed by an output \( o=1 \).'

- **Model** describes the hardware itself.
  - For example as a finite state machine:

```
  i=0    i=1    i=1
巡回
o=0   o=1
i=0
```
Common Criteria Certification

- Assurance Class Development:
  - Use refinement to show that the implementation satisfies its security functional requirements.
  - Gives higher assurance (EAL6).
  - Show that the specification satisfies the (security policy related) requirements.
  - Show that the specification has no inconsistencies.

Diagram:
- Functional Requirement
- Functional Specification
- Design Description
- Implementation Representation
- Policy Model
Security Policy Model

- Functional Requirement
- CTL Computation Tree Logic
- No (Counter Example)

- Functional Specification
- FSM Finite State Machine

Model Checker NuSMV

Yes / No (Counter Example)
SPM – Step by Step

- Temporal Logic Formulas:
  - Identify security policies (sets of Security Functional Requirements)
  - Translate SFRs into temporal logic formulas
  - For all policies that are not relevant for the model argue why they are not relevant.

- Finite State machine:
  - Identify relevant parts of the TOE security functionality (ADV_FSP).
  - Translate the relevant parts of the functional specification into Finite State Machines.

- Model Checker:
  - Use the model checker to verify that the FSM satisfies the Temporal Logic Formulas.
Example – Security IC

- Security Policies:
  - Hardware Access Control
  - Application Management

- Identification and Authentication:
  - FMT_SMF.1.1[APP]: ‘The TSF shall be capable of performing the following management functions:
    Authenticate a user,
    Invalidate the current authentication state based on the functions: reset, … ‘

  eventually(authenticated)

  always(reset -> next(!authenticated))
Example

**Functional Requirement**

- eventually(authenticated)
- always(reset -> next(!authenticated))

**Functional Specification**

- ! auth
- auth
- reset
- authentication

**Model Checker**

NuSMV

Yes

Yes
Conclusions

- Formal modeling leads to new insights into the working of the TOE.
- Helps improve documentation (consistency, completeness, unambiguity).
- Gives higher assurance that the claimed Security Functional Requirements are met by the Target of Evaluation.

’Use of formal methods does not a priori guarantee correctness. However, they can greatly increase our understanding of a system by revealing inconsistencies, ambiguities, and incompletenesses that might otherwise go undetected.’ Ed Clarke and Jeannette Wing