Technical challenges and solutions in Sony FeliCa contactless smartcard EAL6+ evaluation

Hideo Yoshimi
FeliCa Business Division
Professional Solutions Group
Sony Corporation

13th ICCC, Paris, September 2012
Contents

- Introduction
- Project overview
- Technical challenges for ADV in EAL6+
- Our approach toward SPM
- Technical challenges for ADV in EAL7
- Conclusion
Introduction – Sony CC Challenges

- FeliCa is the contactless IC card technology developed by Sony.
- We have been awarded many Common Criteria certificates.
- RC-SA00 will get the world’s first EAL6+ certificate* as a smartcard IC chip with embedded software.

* according to our own research (as of Sep. 4, 2012)

EAL6+  RC-SA00
EAL5+    
EAL4+  RC-S251
EAL4    RC-S860, RC-S940, CXD3715GG/GU-x, RC-S960, RC-S962, RC-S952, RC-S954, RC-S957

Supports AES/DES encryption
Supports DES encryption
Project Overview – EAL6+

- **TOE name**
  - FeliCa Contactless Smartcard IC RC-SA00/1 Series and RC-SA00/2 Series

- **PP conformance**
  - BSI PP 0035 (Security IC Platform PP)

- **EAL**
  - **EAL6+ (ASE_TSS.2)**: Advanced operation mode (AES)
  - **EAL4**: Backward-compatible operation mode (DES)

- **TOE Scope**
  - HW + IC dedicated software + OS

![Diagram of IC Card with layered hardware and software components highlighting TOE scope.](image)
# Technical challenges for ADV in EAL6+

- **5 gaps** between EAL4 and EAL6

<table>
<thead>
<tr>
<th>Assurance class</th>
<th>Assurance family</th>
<th>Assurance components by Evaluation Assurance Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EAL1</td>
</tr>
<tr>
<td>Development</td>
<td>ADV_ARC</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_FSP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_TDS</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_INT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_IMP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_SPM</td>
<td></td>
</tr>
</tbody>
</table>

5 families are upgraded.
Semi-formal description of TSF module and TSFI is required.

We use the following defined description in our design documents:

- **= EQUIVALENT** Denotes equivalency between items. For example: Read_command = Read_parameters
- **+ AND** Denotes a logical AND operation. For example: item1 + item2 + item3
- **[ ] EITHER-OR** Used with | (vertical bar) to present a list of at least two mutually-exclusive options.
- **{ } ITERATIONS OF** Denotes repetitions of an item. The number to the left of the statement denotes the lower limit; the one to the right, the upper limit. For example: 1{ eat_apple }3
- **() OPTIONAL** Denotes optional information. For example: parasol + ( sunglasses )
- **| a logical separator** Separates options in EITHER-OR statements. For example: [ option1 | option2 | option3 ]
**INT**

- **Minimal complexity design required**
  - Set the definition of complexity (Cyclomatic complexity = Number of decisions + 1)
  - Set the acceptable threshold
  - Analyze the complexity of every module and justify the design as being minimal complexity

```c
main(){
  if ( a<b && c<0 ){ ............(a)
    for( i=0; i<7; i++ ){...(b)
      ...
    }
  }
  if ( a<c ){ ............(c)
    ...
  }
}
```

- **Threshold**
<table>
<thead>
<tr>
<th>CC</th>
<th>Type of procedure</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-4</td>
<td>Simple</td>
<td>Low</td>
</tr>
<tr>
<td>5-10</td>
<td>Well structured and stable</td>
<td>Low</td>
</tr>
<tr>
<td>11-20</td>
<td>Complex</td>
<td>Moderate</td>
</tr>
<tr>
<td>21-50</td>
<td>Complex and alarming</td>
<td>High</td>
</tr>
<tr>
<td>&gt;50</td>
<td>Error-prone and extremely troublesome</td>
<td>Very high</td>
</tr>
</tbody>
</table>
IMP

**Complete mapping** required

Mapping between TSF modules and entire source code:

**EAL4, 5**
Partial mapping

**EAL6**
Complete mapping
## SPM (Security Policy Model)

- **Biggest challenge** of EAL6+
  - What is the formal security policy model?
  - What is the formal method?
  - Introduce our pioneering approach for this issue

### New item Requirements

<table>
<thead>
<tr>
<th>New item</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADV_SPM.1.1D</td>
<td>The developer shall provide a formal security policy model for the [assignment : list of policies that are formally modelled].</td>
</tr>
<tr>
<td>ADV_SPM.1.2D</td>
<td>For each policy covered by the formal security policy model, the model shall identify the relevant portions of the statement of SFRs that make up that policy.</td>
</tr>
<tr>
<td>ADV_SPM.1.3D</td>
<td>The developer shall provide a formal proof of correspondence between the model and any formal functional specification.</td>
</tr>
<tr>
<td>ADV_SPM.1.4D</td>
<td>The developer shall provide a demonstration of correspondence between the model and the functional specification.</td>
</tr>
</tbody>
</table>
Our Approach toward SPM

1. Preparation
2. CC requirements
3. Evaluation methodology (AIS34)
4. Our interpretation
5. Modelling approach
6. Semi-formal modelling
7. Formal method choice
8. Model check
9. Formal implementation
10. Verification result
1. Preparation

- Team study
  - Security engineers
  - Ph.D. formal language expert

- Start from the study of formal methods and tools
  - We study the formal methods (SPIN, B, VDM++, etc.)

- CC requirements analysis
  - What to do for the formal SPM
2. CC Requirements

CC requirements

<table>
<thead>
<tr>
<th>New item</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADV_SPM.1.1D</td>
<td>The developer shall provide a formal security policy model for the</td>
</tr>
<tr>
<td></td>
<td>[assignment: list of policies that are formally modelled].</td>
</tr>
</tbody>
</table>

What is the formal security policy model?

CC application notes

A formal security model is a precise formal presentation of the important aspects of security and their relationship to the behaviour of the TOE;

...  

The Security Policy Model of the TOE is informally abstracted from its realisation by considering the proposed security requirements of the ST. The informal abstraction is taken to be successful if the TOE's **principles** (also termed “invariants”) turn out to be enforced by its **characteristics**. The purpose of formal methods lies within the enhancement of the rigour of enforcement. Informal arguments are always prone to fallacies; especially if relationships among subjects, objects and operations get more and more involved. In order to minimise the risk of insecure state arrivals the **rules and characteristics** of the security policy model are mapped to respective **properties and features** within some formal system, whose rigour and strength can afterwards be used to obtain the security properties by means of theorems and formal proof.
3. Evaluation methodology (AIS34)

- The evaluation methodology for EAL6+ is not included in the CEM.
- CEM provides methodology for EAL1 to EAL5.
- **AIS34** extends the CEM to EAL6.

```
ST  SFR selection

Functional specification

Informal SPM

Security characteristics

Security principles

Formal SPM

Security features

Security properties

SFR selection

“consistent and complete”
```

Copyright 2012 Sony Corporation
4. Our interpretation

**Informal SPM**

**Security characteristics**
- FDP_ACC.1 in [ST]
- FDP_ACF.1 in [ST]
- FMT_MSA.1 in [ST]

**Security principles**
- O.AC
  The TOE shall provide a configurable access control to prevent unauthorised access to stored user data.

**Formal SPM**

**Security features**
- abstract state machine processing commands

**Security properties**
- assert(unauthorised access == never succeed)
5. Modelling approach

**Informal SPM**

**Security characteristics**
- FDP_ACC.1 in [ST]
- FDP_ACF.1 in [ST]
- FMT_MSA.1 in [ST]

**Security principles**
- O.AC
  The TOE shall provide a configurable access control to prevent unauthorised access to stored user data.

---

**Semi-formal SPM**

**Functional specification**

"consistent and complete"

---

**Formal SPM**

**Security features**
abstract state machine processing commands

**Security properties**
assert(unauthorised access == never succeed)
6. Semi-formal modelling (1/2)

- Define the behaviour of the TOE
  - State transition model:

    Command 1 → state1 → Command 2 → state2 → Command 3 → state3

    Power off

- File system model:

Security characteristics
- FDP_ACC.1 in [ST]
- FDP_ACF.1 in [ST]
- FMT_MSA.1 in [ST]

Security features
abstract state machine processing commands
6. Semi-formal modelling (2/2)

Define an insecure state of the TOE

Illegal state transition:

- Command 1
  - State 1
  - Command 2
  - State 2
  - Illegal state transition
  - Command 3
  - State 3
  - Illegal state transition

- Illegal data access:
  - Illegal Write access to read-only data
  - Illegal Read access to protected data (without key)

Informal SPM

Security principles
- O.A.C
  - The TOE shall provide a configurable access control to prevent unauthorised access to stored user data.

Semi-formal SPM

Security properties
- assert(unauthorised access == never succeed)
7. Formal Method Choice

Formal method
- Means the model that is applicable to the specification and proof of system.
- "Formal" means it is precise enough to be implemented on a computer.
- With the formal method, we can develop the model of specification and use it as input to a theorem prover.

Approach 1

Proof assistant (Coq, Agda, etc.)
- Mathematics theory is needed
- Now mainly used by mathematician

Approach 2

Model check (SPIN, NuSMV, etc.)
- Exhaustive analysis of all states to verify the behaviour
- Automatic execution by computer
- Easy to use for engineer as well as mathematician

We adopt Approach 2
8. What is Model Check?

- 2 work items for Model Check
  - Create model (**security features**)
  - Create verification logic (**security properties**)

### Semi-formal SPM

```
Command 1
\[\text{state1}\] \rightarrow \[\text{state2}\] \rightarrow \[\text{state3}\]
Power off \[\text{Command 3}\]
```

### Formal SPM

#### Security features

```
chan cmd_line = [1] of \{mtype\};
chan rsp_line = [1] of \{mtype\};
inline cl_auth_v1(){
  command_code = Authentication;
  cmd_line!command_code(0);
}

inline receive_response(transactionkey){
  assert(transactionkey == TOE_TransactionKey &&
         access_mode == DATA);
}
```

#### Security properties

```
inline
receive_response(transactionkey){
  assert(transactionkey ==
         TOE_TransactionKey &&
         access_mode == DATA);
}
```

### Automatic exhaustive analysis

- **[Verification result]**
  - **no error**: Insecure state is unreachable
  - **error**: insecure state is reachable

---

Copyright 2012 Sony Corporation
9. Formal implementation

- **SPIN**
  - Major tool for Model Check
  - Formal language: Promela
  - **Automated exhaustive analysis** execution

Formal language implementation (sample code)

![Sample Code](image)

Total code size: 4.5KLine
10. Verification result

Security verification result by SPIN

All states are exhaustively searched
(errors: 0 means unreachable to an insecure state)
Technical challenges for ADV in EAL7

- **2 gaps** between EAL6 and EAL7
  - FSP & TDS
  - Formal descriptions of TSFI and TSF subsystem are required.

<table>
<thead>
<tr>
<th>Assurance class</th>
<th>Assurance family</th>
<th>Assurance components by Evaluation Assurance Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EAL1</td>
</tr>
<tr>
<td>Development</td>
<td>ADV_ARC</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_FSP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_TDS</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_INT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_IMP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_SPM</td>
<td></td>
</tr>
</tbody>
</table>

2 families are upgraded
Our approach with VDM++

- VDM++ is the major method for formal specification.
- We created the formal description of all TSFI and TSF subsystems.
  - Purpose: not for EAL7, but for software quality improvement
  - Outcome: 17000 LOC, some bugs found

EAL7 point of view
- VDM++ seems to meet the CC requirements for developer evidence.
- At this moment, however, there is no evaluation methodology for EAL7.
## Conclusion

We have shown the technical challenges and our solutions for the world’s first EAL6+ certification.

<table>
<thead>
<tr>
<th>Our approach</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FSP</strong></td>
<td>✔️</td>
</tr>
<tr>
<td>Semi-formal description</td>
<td>✔️</td>
</tr>
<tr>
<td>• use the defined description in our design documents</td>
<td></td>
</tr>
<tr>
<td><strong>TDS</strong></td>
<td>✔️</td>
</tr>
<tr>
<td>Minimal complexity design</td>
<td>✔️</td>
</tr>
<tr>
<td>• analyze the complexity and justify minimal-complexity design</td>
<td></td>
</tr>
<tr>
<td><strong>INT</strong></td>
<td>✔️</td>
</tr>
<tr>
<td>Complete mapping</td>
<td>✔️</td>
</tr>
<tr>
<td>• create mapping between TSF modules and entire source code</td>
<td></td>
</tr>
<tr>
<td><strong>IMP</strong></td>
<td>✔️</td>
</tr>
<tr>
<td>Formal security policy model</td>
<td>✔️</td>
</tr>
<tr>
<td>• create the formal SPM with SPIN and provide the formal proof of non-reachability to an insecure state</td>
<td></td>
</tr>
</tbody>
</table>
“Sony” or “make.believe” is a registered trademark of Sony Corporation. Names of Sony products and services are the registered trademarks and/or trademarks of Sony Corporation or its Group companies. Other company names and product names are the registered trademarks and/or trademarks of the respective companies.