Wouter Slegers

Teaching CC: Lessons learned
When one teaches, two learn.

Robert Heinlein
Teaching the CC: Lessons learned

Progression of training methods

1. Here is the standard
2. Following the standard
3. Big picture + following the standard
4. Big picture
5. Hands-on

Effect of experience and roles

... and what the future might bring
The Common Criteria

The member organisations of the CCRA declare that defined assurance levels (EALs) between versions of the criteria are equivalent and can therefore be used without restrictions for composition activities.

Note CC v3.1. Release 3 now

Click here for information about the CC/CEM maintenance process.

CC v3.1 consists of three parts, make sure to download and use these files marked as "Final":

Part 1: Introduction and general model
- CCPART1V3.1R3.pdf
- CCPART1V3.1R3.dtd

Part 2: Security functional requirements
- CCPART2V3.1R3.pdf
- CCPART2V3.1R3.dtd
- CCPART2V3.1R3 - marked changes.pdf

Part 3: Security assurance requirements
- CCPART3V3.1R3.pdf
- CCPART3V3.1R3.dtd
- CCPART3V3.1R3 - marked changes.pdf

CEM v3.1 consists of one part:

- CEMV3.1R3.pdf
- CEMV3.1R3 - marked changes.pdf

See the new guides:
- Guide for the Transition from CC v2.3 to CC v3.1 for ADV requirements
- Transition Guide: Summary of changes in the CC/CEM from v2.3 to v3.1
- Guidelines for Developer Documentation according to Common Criteria Version 3.1

CC v2.3

CC v2.3 is based on version 2.3, updated with a number of interpretations and further editorial changes.
Complex
Teaching the CC: Lessons learned

Progression of training methods

1. Here is the standard
2. Following the standard
3. Big picture + following the standard
4. Big picture
5. Hands-on

Effect of experience and roles

... and what the future might bring
1000+ pages of confusion

Innocent start

Se-cu-ri-ty
Func-tio-nal
Re-eh-qui-rement

FCS_COP.1(RSA)

What exactly is SFR-supporting?

What really is “consistent”?

How do I get the CB not to bother me again?

How do I get the project manager of my back?

How do I get the developer to really fix it this time?

When is it good enough to approve?

How do I convince the CB?

How to efficiently evaluate as a lab?

The CC makes no sense!

The CC has little sense

... Except for what I put into it
1000+ pages of confusion

Innocent start

Se-cu-ri-ty

Func-tio-nal

Re-eh-qui-rement

FCS_COP.1(RSA)

What really is “consistent”?

What exactly is SFR-supporting?

How do I get the CB not to bother me again?

How do I get the project manager of my back?

How do I get the developer to really fix it this time?

When is it good enough to approve?

How do I convince the CB?

The CC makes no sense!

The CC has little sense

How to efficiently evaluate as a lab?

...Except for what I put into it

Why does it fail? Why does it pass? How to optimize?
Following the standard

Advantages:
- Everything has been told
- Good terminology coverage

Disadvantages:
- Little is remembered
- Mindset of failing the TOE/IR

Typical:
- 2-5 day training
- Students “survived it”
Teaching the CC: Lessons learned

Progression of training methods

1. Here is the standard
2. Following the standard
3. Big picture + following the standard
4. Big picture
5. Hands-on

Effect of experience and roles

... and what the future might bring
Big picture overview before standard

Advantages:
- Structure to make sense of standard
- Good recall and integration

Disadvantages:
- Too much knowledge
- Students get frustrated/sassy with standard course trainers

Typical:
- 4h-1d training (big picture) + ...
Teaching the CC: Lessons learned

Progression of training methods

1. Here is the standard
2. Following the standard
3. Big picture + following the standard
4. Big picture
5. Hands-on

Effect of experience and roles

... and what the future might bring
Big picture overview

Advantages:
• Structure to learn from
• Matches realistic learning goals

Disadvantages:
• Poor coverage of the terminology
• Not practically applicable directly

Typical:
• 1h-1d training
• Great for management level
Teaching the CC: Lessons learned

Progression of training methods

1. Here is the standard
2. Following the standard
3. Big picture + following the standard
4. Big picture
5. Hands-on

Effect of experience and roles

... and what the future might bring
Why does it fail? 

Why does it pass? 

How to optimize?

What really is “consistent”?

How do I get the CB not to bother me again?

How do I get the project manager of my back?

How do I get the developer to really fix it this time?

When is it good enough to approve?

How do I convince the CB?

The CC makes no sense!

The CC has little sense

...Except for what I put into it

FCS_COP.1(RSA)

Se-cu-ri-ty

Func-tio-nal

Re-ch-qui-rement

1000+ pages of confusion

Innocent start
Hands-on: showing what to do
### Hands-on: showing what to do

<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCS_COP.1</td>
<td>Encryption/decryption</td>
<td>A</td>
<td>TSFI 1</td>
<td>Subsystem A encrypts/decrypts based on the direction of the communication: inbound messages are decrypted, outbound are encrypted.</td>
<td>Encryption is tested in the test case “Encryption check”, decryption in the test case “Decryption check”.</td>
</tr>
<tr>
<td>(TEA)</td>
<td></td>
<td></td>
<td>[REF1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tiny Encryption Algorithm</td>
<td>A</td>
<td>See above, algorithm identifier for TEA is 0x42</td>
<td>TEA is implemented in subsystem A.</td>
<td>In the test cases listed above, the algorithm is listed under that value.</td>
<td></td>
</tr>
<tr>
<td>(TEA)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>128 bits</td>
<td>A</td>
<td>See above</td>
<td></td>
<td>TEA is only possible with 128 bit key sizes, and is implemented in subsystem A.</td>
<td>In the test cases listed above, this key size is used.</td>
</tr>
</tbody>
</table>
## ST: How the SFRs are implemented

<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
</table>
| FCS_COP.1 (TEA) | Encryption/decryption     | A            | TSFI 1 [REF1] | Subsystem A encrypts/decants based on the direction of the communication: inbound messages are decrypted, outbound are encrypted. | Encryption is tested in the test case “Encryption check”, decryption in the test case “Decryption check“.
|                |                           |              |            |                                                                            |                                                                            |
| Tiny Encryption Algorithm (TEA) |                      | A            | See above, algorithm identifier for TEA is 0x42 | TEA is implemented in subsystem A.                                           | In the test cases listed above, the algorithm is listed under that value. |
| 128 bits       |                           | A            | See above  | TEA is only possible with 128 bit key sizes, and is implemented in subsystem A. | In the test cases listed above, this key size is used. |
## ADV_FSP: In what TSFI the SFRs are implemented

<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
</table>
| FCS_COP.1 (TEA)   | Encryption/decryption           | A            | TSFI 1 [REF1] | Subsystem A encrypts/decrypts based on the direction of the communication: inbound messages are decrypted, outbound are encrypted. | Encryption is tested in the test case “Encryption check”, decryption in the test case “Decryption check“.
<p>| | | | | | |
|                   |                                 |              |          |                                                                           |                                                                              |
| Tiny Encryption Algorithm (TEA) |                                  | A            | See above, algorithm identifier for TEA is 0x42 | TEA is implemented in subsystem A. | In the test cases listed above, the algorithm is listed under that value. |
| 128 bits          |                                 | A            | See above | TEA is only possible with 128 bit key sizes, and is implemented in subsystem A. | In the test cases listed above, this key size is used. |</p>
<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCS_COP.1 (TEA)</td>
<td>Encryption/decryption</td>
<td>A</td>
<td>TSFI 1</td>
<td>Subsystem A encrypts/decrypts based on the direction of the communication: inbound messages are decrypted, outbound are encrypted.</td>
<td>Encryption is tested in the test case “Encryption check”, decryption in the test case “Decryption check”.</td>
</tr>
<tr>
<td>Tiny Encryption Algorithm (TEA)</td>
<td>See above, algorithm identifier for TEA is 0x42</td>
<td>A</td>
<td>See above</td>
<td>TEA is implemented in subsystem A.</td>
<td>In the test cases listed above, the algorithm is listed under that value.</td>
</tr>
<tr>
<td>128 bits</td>
<td></td>
<td>A</td>
<td>See above</td>
<td>TEA is only possible with 128 bit key sizes, and is implemented in subsystem A.</td>
<td>In the test cases listed above, this key size is used.</td>
</tr>
</tbody>
</table>
## ATE: How the SFRs are tested

<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
</table>
| FCS_COP.1 (TEA)     | Encryption/decryption          | A            | TSFI 1   | Subsystem A encrypts/decrypts based on the direction of the communication: inbound messages are decrypted, outbound are encrypted. | Encryption is tested in the test case “Encryption check”, decryption in the test case “Decryption check“.
| Tiny Encryption Algorithm (TEA) |                        | A            | See above, algorithm identifier for TEA is 0x42 | TEA is implemented in subsystem A.                                      | In the test cases listed above, the algorithm is listed under that value. |
| 128 bits            |                                | A            | See above | TEA is only possible with 128 bit key sizes, and is implemented in subsystem A. | In the test cases listed above, this key size is used.                      |
## Review

<table>
<thead>
<tr>
<th>SFR</th>
<th>SFR detail</th>
<th>Subsystem(s)</th>
<th>TSFI</th>
<th>Rationale</th>
<th>Test case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCS_COP.1 (TEA)</td>
<td>Encryption/ decryption</td>
<td>A</td>
<td>TSFI 1 [REF1]</td>
<td>Subsystem A encrypts/decrypts based on the direction of the communication: inbound messages are decrypted, outbound are encrypted.</td>
<td>Encryption is tested in the test case “Encryption check”.</td>
</tr>
<tr>
<td>Tiny Encryption Algorithm (TEA)</td>
<td>A</td>
<td>See above, algorithm identifier for TEA is 0x42</td>
<td>TEA is implemented in subsystem A.</td>
<td>In the test cases listed above, the algorithm is listed under that value.</td>
<td></td>
</tr>
<tr>
<td>128 bits</td>
<td>A</td>
<td>See above</td>
<td>TEA is only possible with 128 bit key sizes, is implemented in subsystem A.</td>
<td>In the test cases listed above, this key size is used.</td>
<td></td>
</tr>
</tbody>
</table>
Hands-on

Advantages:
• Immediately effective to work 90+%  
• Implicit framework anchored in work

Disadvantages: ?

Tricks:
• Not telling them this is hard
• Telling them what to do  
• ... (and when to call help)
• Sneaking in the mindset
Innocent start

1000+ pages of confusion

Security

Functional

Requirement

FCS_COP.1(RSA)

What exactly is SFR-supporting?

What really is “consistent”?

How do I get the CB not to bother me again?

How do I get the project manager of my back?

How do I get the developer to really fix it this time?

When is it good enough to approve?

How do I convince the CB?

How to efficiently evaluate as a lab?

The CC makes no sense!

The CC has little sense

... Except for what I put into it

Why does it fail? Why does it pass? How to optimize?
Effect of roles on further growth

No practical application:
- Knowledge fades
Different roles different views

- Consultant
- Evaluator
- Trainer
- Certifier
- Project manager
1 Role

Typical:

• Consultant, or
• Evaluator, or
• Trainer
• Certifier
• Project manager
Cross training in 2+ roles

Typical:
- Consultant, and
- Evaluator
- Trainer
- Certifier
- Project manager
Cross training effects can be captured

Relevant for:

- Consultant
- Evaluator
- Trainer
- Certifier
- Project manager
Skill: intellectual integrity

1. “Be honest to yourself and others”

2. “Honor the rules of the game”

3. “Don't invent new rules yourself”
The only valid measurement of code quality: WTFs/minute

Good code.

Bad code.

(c) 2008 Focus Shift/0Shews/Thom Holwerda - http://www.osnews.com/comics
My favorite: hands-on + skill training
Teaching the CC: Lessons learned (1)

Beginning in CC:

1. Do not tell them this is hard
2. Show them practical steps to do
3. ... and when to get senior help
4. Get them running and gathering experience
Teaching the CC: Lessons learned (2)

Experienced in CC:

1. Cross-train in roles
2. Teach the skills:
   1. Reasoning,
   2. Interpersonal relating,
   3. Communication,
   4. Keeping integrity
   5. (and security view)
3. Relax around what does not matter
Teaching the CC: Future?

- CC for professionals: evaluation processes
  http://www.yourcreativesolutions.nl/books/

- More?
Wouter Slegers

CREATIVE SOLUTIONS

WOUTER@YOURCREATIVESOLUTIONS.NL
+31-6-24721971