CS42600: Computer Security

Assurance

Prof. Chris Clifton

16 April 2019

Center for Education and Research in Information Assurance and Security

Common Criteria

- International standard for evaluating assurance
- CC Documents
  - Functional requirements
  - Assurance requirements
  - Evaluation Assurance Levels
- CC Evaluation Methodology
  - Detailed process model for each level
- National Scheme

48

PURDUE UNIVERSITY

Common Criteria: Origin

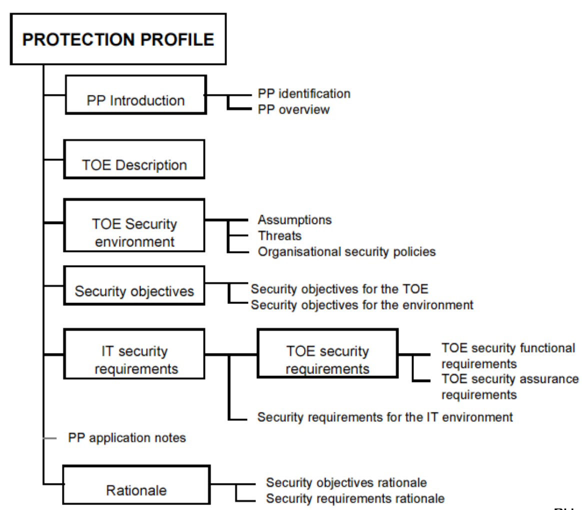
49

PURDUE UNIVERSITY

Common Criteria: Protection Profile

Domain-specific set of security requirements

- Narrative Overview
- Domain description
- Security Environment (threats, overall policies)
- Security Objectives: System, Environment
- IT Security Requirements
  - Functional drawn from CC set
  - Assurance level
- Rationale for objectives and requirements



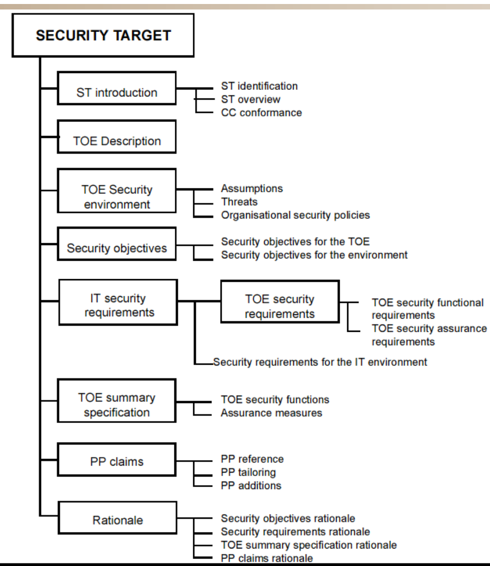
50

PURDUE UNIVERSITY

Common Criteria: Security Target

Specific requirements used to evaluate system

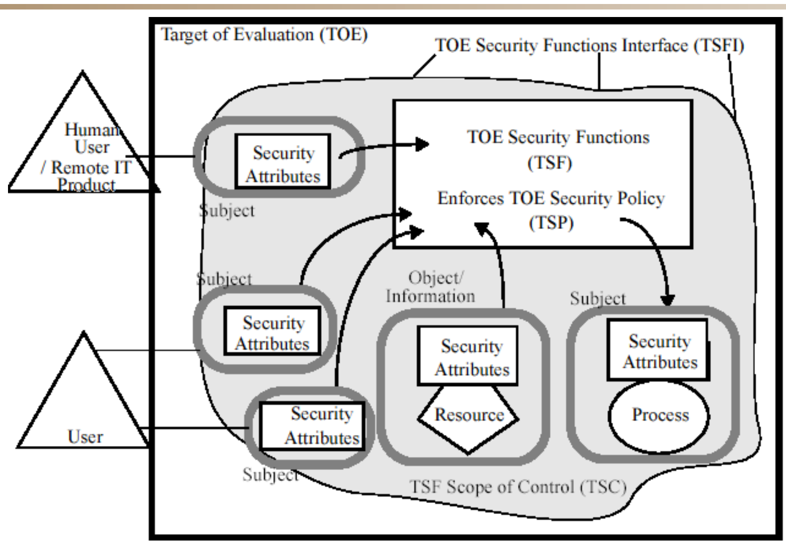
- Narrative introduction
- Environment
- Security Objectives
  - How met
- Security Requirements
  - Environment and system
  - Drawn from CC set
- Mapping of Function to Requirements
- Claims of Conformance to Protection Profile



51

PURDUE UNIVERSITY

Security Paradigm



52

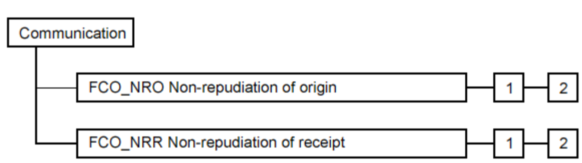
PURDUE UNIVERSITY

Common Criteria: Functional Requirements

- 676 page document
  - Plus 430 pages for evaluation methodology
- 17 Classes
  - Audit, Communication, Cryptography, User data protection, ID/authentication, Management, Privacy, Protection of Security Functions, Resource Utilization, Access, Trusted paths
- Several families per class
- Lattice of components in family

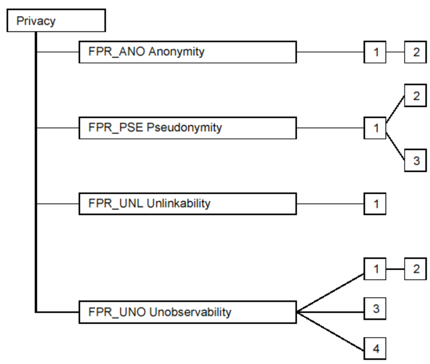
PURDUE UNIVERSITY

Class Example: Communication



- Non-repudiation of origin
  1. Selective Proof. Capability to request verification of origin
  2. Enforced Proof. All communication includes verifiable origin

Class Example: Privacy



1. Pseudonymity
   1. The TSF shall ensure that [assignment: *set of users and/or subjects*] are unable to determine the real user name bound to [assignment: *list of subjects and/or operations and/or objects*]
   2. The TSF shall be able to provide [assignment: *number of aliases*] aliases of the real user name to [assignment: *list of subjects*]
   3. The TSF shall [selection: *determine an alias for a user, accept the alias from the user*] and verify that it conforms to the [assignment: *alias metric*]
2. Reversible Pseudonimity
   1. …
3. Alias Pseudonimity
   1. …

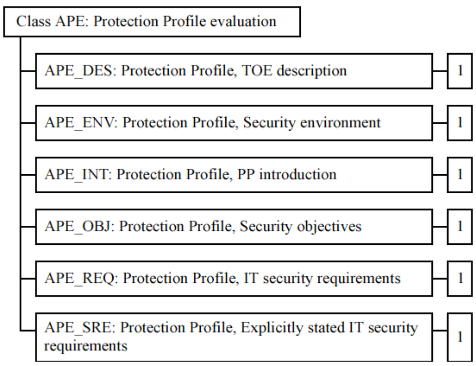
55

Common Criteria: Assurance Requirements

- 247 page document
- 7 Evaluation Levels
- 9 Classes
  – Protection Profile Evaluation, Configuration Evaluation, Security Target Evaluation
  – Development, Guidance, Life cycle, Tests, Vulnerability assessment
  – Composition
- Several families per class
- Lattice of components in family

56

Example:
Protection Profile Evaluation



Security environment

- In order to determine whether the IT security requirements in the PP are sufficient, it is important that the security problem to be solved is clearly understood by all parties to the evaluation.

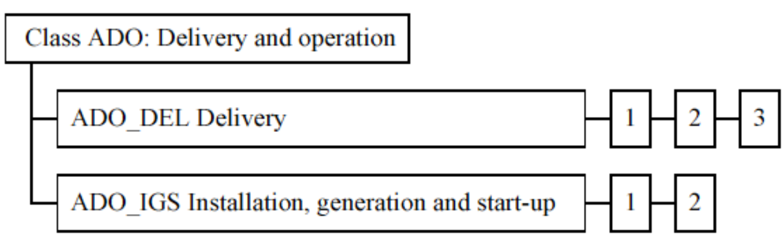
1. Protection Profile, Security environment, Evaluation requirements
   - Dependencies: No dependencies.
   - Developer action elements:

- The PP developer shall provide a statement of TOE security environment as part of the PP.
  - Content and presentation of evidence elements:
- The statement of TOE security environment shall identify and explain any assumptions about the intended usage of the TOE and the environment of use of the TOE.
- The statement of TOE security environment shall identify and explain any known or presumed threats to the assets against which protection will be required, either by the TOE or by its environment.
- The statement of TOE security environment shall identify and explain any organisational security policies with which the TOE must comply.
  - Evaluator action elements:
- The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
- The evaluator shall confirm that the statement of TOE security environment is coherent and internally consistent.

57

PURDUE
UNIVERSITY

Example:
Delivery and Operation



Installation, generation and start-up

A. Installation, generation, and start-up procedures
   - Dependencies: AGD_ADM.1 Administrator guidance

B. Developer action elements:
   - The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

C. Content and presentation of evidence elements:
   - The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

D. Evaluator action elements:
   - The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
   - The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

Generation Log

58

Common Criteria: Evaluation Assurance Levels

1. Functionally tested
2. Structurally tested
3. Methodically tested and checked
4. Methodically designed, tested, and reviewed
5. Semiformally designed and tested
6. Semiformally verified design and tested
7. Formally verified design and tested

59

Common Criteria: Evaluation Process

- National Authority authorizes evaluators
  - U.S.: NIST accredits commercial organizations
  - Fee charged for evaluation
- Team of four to six evaluators
  - Develop work plan and clear with NIST
  - Evaluate Protection Profile first
  - If successful, can evaluate Security Target

60

PURDUE UNIVERSITY

Common Criteria: Status

- Over 2000 registered products
  - Seven at level 7 and above
    “Data Diode”, Optical Switch, VM
  - IBM OSs at Level 5
  - Windows, Red Hat Linux Level 4
  - Likely many more not registered
- http://commoncriteriaportal.org

61

PURDUE UNIVERSITY

Next: Formal Methods

- Software verification beyond scope of course
  - But important to achieve security
- Limited software verification
  - Verify the security subsystems
  - *Confine* the rest

62

PURDUE UNIVERSITY

Formal Verification: Components

- Formal Specification defined in unambiguous (mathematical) language
  - Example: security policy models
- Implementation Language
  - Generally somewhat constrained
- Formal Semantics relating the two
- Methodology to ensure implementation ensures specifications met

63

PURDUE UNIVERSITY

Specification Languages

- Specify WHAT, not HOW
  - Valid states of system
  - Postconditions of operations
- Non-Procedural
- Typical Examples:
  - Propositional / Predicate Logic (see Chapter 34)
  - Temporal Logic (supports before/after conditions)
  - Set-based models (e.g., formal Bell-LaPadula model of 5.2.3)

64

PURDUE UNIVERSITY

Specification Languages

- Must support machine processing
  - Strong typing
  - Model input/output/errors
- Example: SPECIAL
  - First order logic base
  - Strongly typed
  - VFUN: describes variables (state)
  - OFUN: describe state transitions

PURDUE UNIVERSITY

Example: SPECIAL

if $(r \notin \Delta(\rho_6))$
  then $\rho_6(r,v) = (\mathbf{i}, v)$

else if $([\, o \neq root(o)$ **and** $parent(o) \neq root(o)$ **and** $parent(o) \in b(s_1\!: \mathbf{w}) \,]$ **or**
  $[parent(o) = root(o)$ **and**
  $canallow(s_1, o, v) \,]$ **or**
  $[o = root(o)$ **and** $canallow(s_1, root(o), v) \,])$

then $\rho_6(r,v) = (\mathbf{y}, (b, m + m[s_2,o] \leftarrow \mathbf{r}, f, h))$

else $\rho_6(r,v) = (\mathbf{n}, v)$

```
MODULE Bell_LaPadula_Model Give_read
Types
Subject_ID:  DESIGNATOR;
Object_ID:  DESIGNATOR;
Access_Model:  {READ, APPEND, WRITE};
Access:  STRUCT_OF(Subject_ID subject;
    Object_ID object; Access_Mode mode);
Functions
VFUN active (Object_ID object) -> BOOLEAN
    active:  HIDDEN; INITIALLY TRUE;
VFUN access_matrix() -> Access accesses:
    HIDDEN;
    INITIALLY FORALL Access a: a
    INSERT accesses => active(a.object);
OFUN give_access(Subject_ID giver; Access
    access);
    ASSERTIONS active(access.object) =
    TRUE;
    EFFECTS `access_matrix() =
    access_matrix() UNION (access);
END_MODULE
```

Verification Methodologies

- Proof based vs. model based
  - Proof: Formula define premises / conclusions
    - Proof shows how to reach conclusions from premises
  - Model-based: Premises and conclusions have compatible truth tables
- Full vs. property verification
  - Does methodology model full system?
  - Or just prove certain key properties?
- Automation – may be manual or have tool support

67

Example: Enhanced Hierarchical Development Methodology

- Proof-based method
  - Uses Boyer-Moore Theorem Prover
- Hierarchical approach
  - *Abstract Machines* defined at each level
    - specification written in SPECIAL
  - *Mapping Specifications* define functionality in terms of machines at higher layers
  - *Consistency Checker* validates mappings "match"
- Compiler that maps a program into a theorem-prover understood form
- Successfully used on MLS systems
  - Few formal policy specifications outside MLS domain

68

Alternate Approach: Combine Specifications and Language

- Specifications defined on procedures
  - Entry conditions
  - Exit conditions
  - Assertions
- Proof techniques ensure exit conditions / assertions met given entry conditions
  - Also run-time checking
- Examples:
  - Gypsy (in book) – uses theorem prover
  - CLU
  - Eiffel (and derivatives) – run-time checks

Other Examples

- Prototype Verification System (PVS)
  - Based on EHDM
  - Interactive theorem-prover
- Symbolic Model Verifier
  - Temporal logic based
  - Notion of "path" – program represented as tree
  - Statements that condition must hold at *a* future state, *all* future states, all states on one path, etc.

Is this Real?

- Formal verification of protocols
  - Key management
  - Protocol development
- Verification of libraries
  - Entire system not verified
  - But components known okay
- High risk subsystems

71