

SML Tutorial

Waël Hassan
University of Ottawa

wael@acm.org

© 2009



what is SML? silver

- A formal interface language that assists security model designers to create Alloy models
- SML is relational, it includes patterns to specific to security
- SML is compact, can be used for quick prototyping, yet learning SML does not preclude a designer from learning Alloy



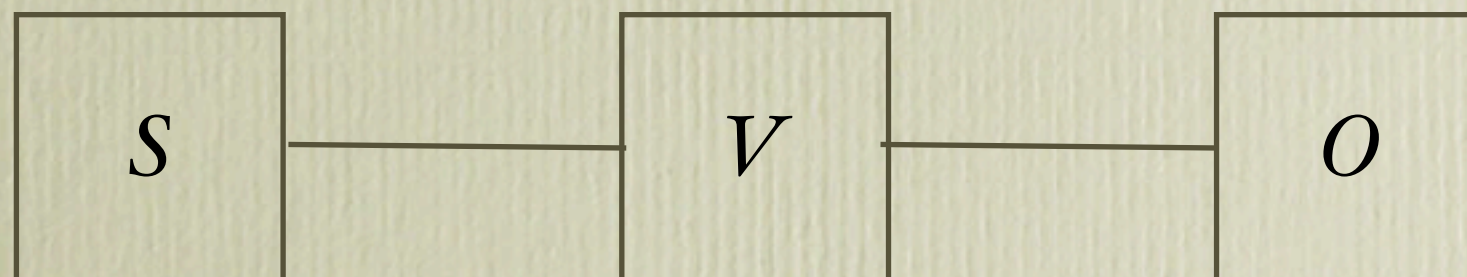
Silver

Silver is type of an Alloy



Designing Secrecy model

- A secrecy model can be represented using a set of classes and relations
- Flat Access model, for example, proposes three classes: Subjects, Actions or Verbs, and objects can be represented using three classes with two relations



SVO Model

- Classes: Subject, Verb, Object
- Relations:
 - A subject can perform a verb
 - A verb operates over an object

SVO in SML

- `Classes(subject,verb,object)`
- `Relation(subject,verb,perform,some)`
- `Relation(verb,object,over,some)`

Abstract Classes

- Define non-instantiated classes
- AabstractClasses (Class-1, Class-2, .., Class-n)
- Example:
 - classification is an abstract class in a multi-level secrecy model

Classes

- identify basic constructs, each construct may have attributes
- $\text{Classes}(\text{Class-1}, \text{Class-2}, \dots, \text{Class-n})$

Example:

- Top-Security is a class in a multi-level security system
- Role is a class in a role dependent environment

Elements

- are attributes of a class
- a subject can have a name
 - `Element(subject, name)`

Extends

- defines sub-classes
- `extends(class-name-child, class-name-parent)`
- Example:
 - `extends(secret, classification)`
 - A secret class extends classification an abstract class

Relation

- is a relation between two classes.
- `Relation(class-1,class-2,Relation-Name,cardinality)`
- Cardinality = { one,some}
- Example
 - `Relation(subject, verb, can, some)`
 - A subject has a relationship 'can' with 'some' verbs

Separate

- Separate is a kind of a relation
- Separate acts on classes with respect to a relation.
- $\text{Separate}(\text{class-1}, \text{class-2}, \text{relation-r})$
- Separate class-1 from class-2 with respect to relation. This separates all classes that have a relation-r with class-1 to have a similar relation to class-2

Separate

- Separate(Internet,Disk,Write)
- Separate the classes internet and disk with respect to write access. The consequences are that those classes that relate using 'write'

Implication

- Implication creates a one way relationship between relations
- $\text{Implication}(\text{relation-1}, \text{relation-2})$
- classes that are joined by relation-1 are also joined by relation-2, not necessarily reciprocal

Equivalence

- Equivalence creates a binding between relations
- `Equivalence(relation-1, relation-2)`
- classes that are joined by relation-1 are also joined by relation-2 and vice-versa