Model-based Automated Security Functional Testing

Dr. R. Chandramouli (Mouli)
mouli@nist.gov

(Security Division – National Inst. Of Standards & Technology

OMG's Seventh Annual Workshop On

DISTRIBUTED OBJECTS and COMPONENTS SECURITY

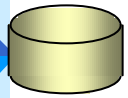
April 7-10, 2003

Baltimore, Maryland, USA

Common Symbols, Abbreviations and Acronyms

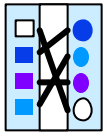
Tool

Manual process

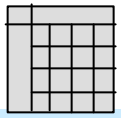


Machine readable artifact

Textual document



Object mapping



Tabular model

API	**Application programming interface**
DCP	**Domain Convergence Path**
GUI	**Graphical user interface**
Java	**High-level programming language**
JDBC	**Java Database Connectivity**
NIST	**National Institute of Standards and Technology**
NRL	**Naval Research Laboratory**
SCR	**Software Cost Reduction**
SFT	**Security Functional Testing**
SQL	**Structured Query Language**
TAF	**Test Automation Framework**
TTM	**T-VEC Tabular Modeler**

Model-based Automated Security Functional Testing (TAF-SFT Toolkit) – Presentation Topics

- Security Testing – Characteristics
- Improving the Economics of Security Functional Testing (TAF)
- TAF for Security Functional Testing (SFT) – TAF-SFT Tool Kit
- TAF-SFT Toolkit – Architecture & Key Process Steps
- TAF-SFT Reference Implementation – Commercial DBMS
- Advantages, Disadvantages & Conclusion

Security Testing - Characteristics

Traditional Software Conformance Testing	*Security Testing*
Verification of Correctness – Market determines Effectiveness	Both Correctness & Effectiveness are integral parts of specifications
Verification for Conformance to Functional Specs	Verification for Conformance to Functional Specs & Underlying Security Model
Statistical Coverage Measures guarantee correct functional behavior	Potential for exploiting obscure flaws to subvert intended security behavior

Security Testing – Characteristics (contd..)

- Two General Categories

1. Security Functional Testing *(WHAT SHOULD DO)*

   – Testing for Conformance to Security Function Specifications & Underlying Security Model

2. Security Vulnerability Testing(*WHAT SHOULD NOT DO)*

   -- Identification of flaws in design or implementation that can subvert intended security behavior

Improving the Economics of Security Functional Testing (TAF)

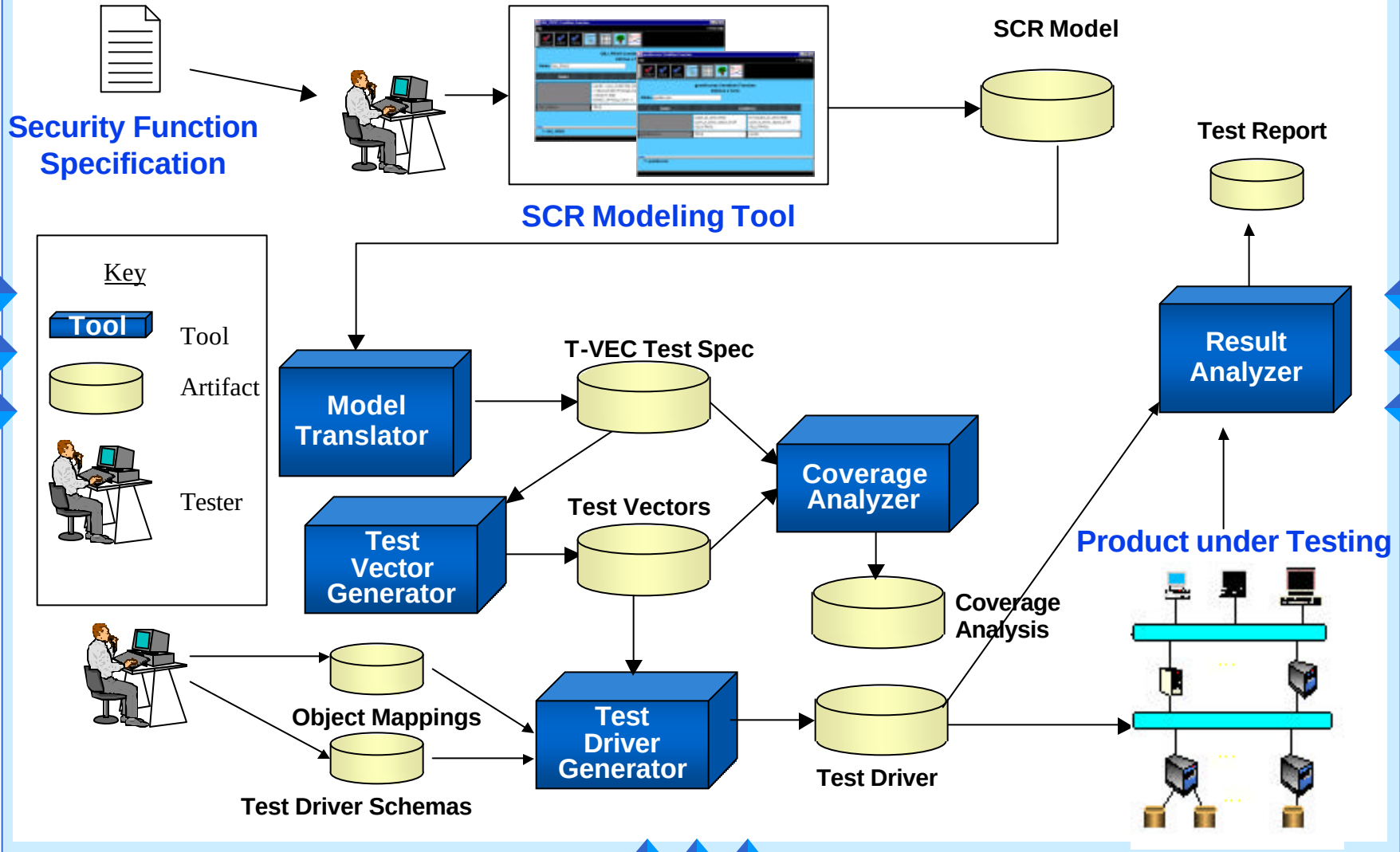
- Independent Security Functional Testing rarely performed in traditional security evaluations & certifications.
  - Complexity: Representing Security Functional specifications & determining coverage
  - Costs: Non-reusability of previously developed tests

- Test Automation Framework (TAF) – Improving the economics of Security Functional Testing through end-to-end tool support.

TAF for Security Functional Testing – TAF-SFT Toolkit (☺ - automated)

- Step 1: Develop a behavioral model of security function specification using a tabular specification language called SCR.
- Step 2:Translate SCR specifications to T-VEC Test Specifications ☺
- Step 3:Generate test vectors from transformed SCR specifications and perform coverage analysis ☺
- Step 4: Develop test driver schemas and object mappings (explained latter) for target test environment.
- Step 5:Generate test drivers, execute tests and generate test report. ☺

TAF-SFT Toolkit - Architecture

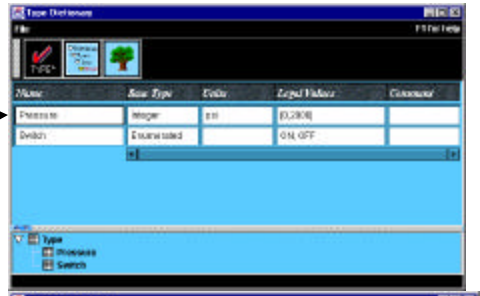


Modeling Security Functions in SCR

Product Security Function Specifications

Modeling Structures

Data Types



Variables

Behavior

State Machine (Modes)

Events

Conditions

SCR Modeling

• Models the behavior of a software system using Tabular functions involving the following types of variables

Variable Class		Description
Controlled		Output object
Monitored		Input object
Term		Auxiliary Variable (Combination of Monitored Variables or other terms)
Mode Class (finite state machine)		Members are Modes. A mode represents a system state

SCR Modeling

- The following are the various Tabular Functions in SCR

Type of Function		Description
Condition		value of a variable under all possible states
Event		the value of a variable after an event occurs
Mode Transition		Shows the source mode, an event and the destination mode

SCR Modeling

(Condition Function Table for Term Variable – *User_Has_Delete_Access*

Table Name	Condition	
	(User_Object_Priv = ‘ALL’) OR (User_Object_Priv = ‘DELETE’)	(User_Object_Priv != ‘ALL’) AND (User_Object_Priv != ‘DELETE’)
User_Has_Delete_Access	TRUE	FALSE

SCR Modeling

(Condition Function Table for Controlled Variable – *Grant_Delete_Access*

Table Name	Condition	
	(UserID=Active_User) AND (User_Has_Delete _Access)	(UserID != Active_User) OR NOT(User_Has_Delete_Access)
Grant_ Delete_ Access	TRUE	FALSE

Translating SCR Model to T-VEC Test Specification

- The T-VEC test specification is made up of
  - Input-Output Functional Relationships
  - Relevance Predicate (a set of constraints on inputs)

- Input-Output Functional Relationship that corresponds to Condition Function Table – Grant_Delete_Access

e.g. (UserID = Active_User) & (User_Has_Delete_Access) → Grant_Delete_Access

Translating SCR Model to T-VEC Test Specification (contd…)

• Relevance Predicate are expressed as

- a set of disjunctions of conjunctions and each disjunction is called a "**DOMAIN CONVERGENCE PATH (DCP)"**

• Relevance Predicates for the Functional Relationship

(UserID = Active_User) & (User_Has_Delete_Access) →

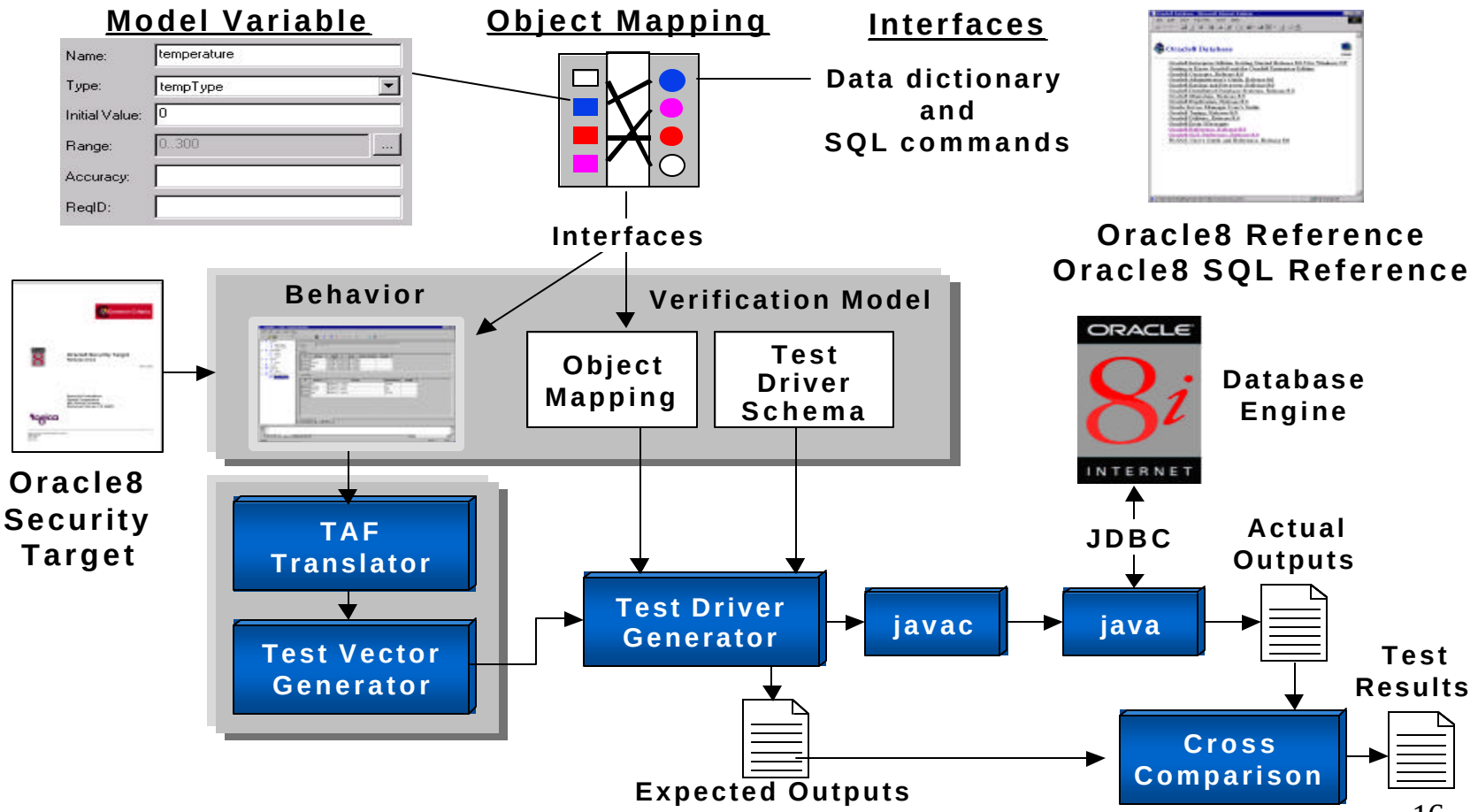
Grant_Delete_Access is

((UserID = Active_User) & (User_Object_Priv = 'ALL'))

OR

((UserID = Active_User) & (User_Object_Priv = 'DELETE'))

Application of TAF-SFT Toolkit for Oracle DBMS Security Functional Testing



Building a SCR Model for a Security Function

• Text Description of the Grant Object Privilege (GOP) Security Function

A normal user (the grantor) can grant an object privilege to another user, role or PUBLIC (the grantee) only if:
a) the grantor is the owner of the object; (GOP (a)) or
b) the grantor has been granted the object privilege with the GRANT OPTION. (GOP (b))

• The interface-related information (SQL commands & valid values) required are: *GRANT <object_privilege> ON <object> TO <user | role | PUBLIC> [WITH GRANT OPTION]* - *where <object_privilege> can be one of: ALL, UPDATE, SELECT,INSERT,DELETE and the GRANT OPTION is optional*

Condition Function Tables for Grant Object Privilege (GOP) Security Function

Table Name	Condition	
	grantor = selectedObjOwner	NOT(grantor = selectedObjOwner)
grantor_owns_object =	TRUE	FALSE

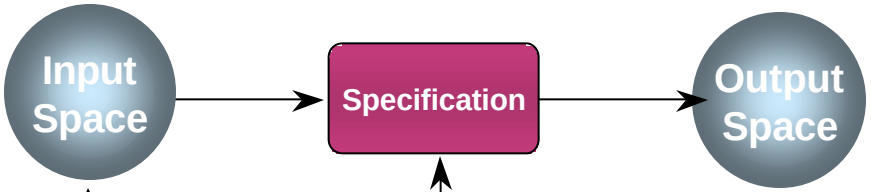
Table Name	Condition		
	(GRANT_OPTION AND selectedObjPriv = grantedObjPriv) AND selectedObj = grantedObj AND selectedObjOwner != grantor AND selectedObjOwner != grantee	NOT(GRANT_OPTION AND selectedObjPriv = grantedObjPriv) AND selectedObj = grantedObj AND selectedObjOwner != grantor AND selectedObjOwner != grantee	**DAC Constraints**
has_grantable_obj_privs =	TRUE	FALSE	

Table Name	Condition		
	((grantor_owns_object) OR (has_grantable_obj_privs)) AND (grantor != grantee) AND (granteeType = user OR (granteeType = role AND granteeRoleID = valid_roleID) OR granteeType = PUBLIC) AND (selectedObjPriv = ALL OR selectedObjPriv = UPDATE OR selectedObjPriv = SELECT OR selectedObjPriv = INSERT OR selectedObjPriv = DELETE)	(NOT(grantor_owns_object)) AND (NOT(has_grantable_obj_privs)) AND (grantor != grantee) AND (granteeType = user OR (granteeType = role AND granteeRoleID = valid_roleID)) AND (selectedObjPriv = ALL OR selectedObjPriv = UPDATE OR selectedObjPriv = SELECT OR selectedObjPriv = INSERT OR selectedObjPriv = DELETE)	**GOP(a)** **GOP(b)** **Domain Constraints**
grant_obj_priv_OK =	TRUE	FALSE	

Converting SCR Specification of GOP Security Function to T-VEC Test Spec

scr2tvec
Model Translator

T-VEC
Test Spec



Relevance Predicate

Input-Output
Functional Relationships

((grantor_owns_object) OR
(has_grantable_object_privs)) AND (<domain constraints>) → grant_obj_priv_OK

((grantor_owns_object) AND (grantee='user') AND (selectedObjPriv = 'UPDATE')) ***OR***

((grantor_owns_object) AND (grantee='PUBLIC') AND (selectedObjPriv = 'UPDATE')) ***OR***

……..

Test Vectors Generated for GOP Security Function

#	TSP	grant_obj_priv_OK	grantor	grantee	granteeType	granteeRoleID	valid_roleID	selectedObjPriv	objOwner	GRANT_OPTION	grantedObjPriv	selectedObj	grantedObj
1	1	TRUE	1	2	user	2	2	ALL	1	TRUE	ALL	4	4
2	1	TRUE	4	3	user	1	1	ALL	4	FALSE	SELECT	1	1
3	2	TRUE	1	2	user	2	2	UPDATE	1	TRUE	ALL	4	4
4	2	TRUE	4	3	user	1	1	UPDATE	4	FALSE	SELECT	1	1
5	3	TRUE	1	2	user	2	2	SELECT	1	TRUE	ALL	4	4
6	3	TRUE	4	3	user	1	1	SELECT	4	FALSE	SELECT	1	1
7	4	TRUE	1	2	user	2	2	INSERT	1	TRUE	ALL	4	4
8	4	TRUE	4	3	user	1	1	INSERT	4	FALSE	SELECT	1	1
9	5	TRUE	1	2	user	2	2	DELETE	1	TRUE	ALL	4	4
10	5	TRUE	4	3	user	1	1	DELETE	4	FALSE	SELECT	1	1
▪ ▪ ▪													
77	39	FALSE	1	2	role	1	1	INSERT	3	FALSE	ALL	1	1
78	39	FALSE	4	3	role	2	2	INSERT	2	FALSE	SELECT	4	4
79	40	FALSE	1	2	role	1	1	DELETE	3	FALSE	ALL	1	1
80	40	FALSE	4	3	role	2	2	DELETE	2	FALSE	SELECT	4	4

Object Mapping & Test Driver Schema

- Object Mapping File
  - Mapping from Model Variables to Interfaces of the System under test (for Oracle 8.0.5 – the interfaces are JDBC Commands, SQL Commands & Oracle Data Dictionary Views)
- Test Driver Schema – Algorithmic pattern for conducting tests

```
Global init;
Forall tests
  init target;
  set inputs;
  execute Test;
  get outputs;
  store output;
endforall
```

TAF-SFT Toolkit Approach – Advantages & Disadvantages

Advantages

- Better Quality of Specifications and quality of test data
- Automated coverage analysis, generation of test code and results analysis

Disadvantages

- Detailed knowledge of security function semantics required for the modeler
- Development of Object Mapping information laborious for products with complex interfaces

Conclusions

Ideal Situations for Maximizing the Return on Investment for TAF-SFT

- Partial re-use of SCR security behavioral model possible
- Partial re-use of Object Mapping Information

Found in Product Environments

- Interoperable security APIs like CDSA and some crypto APIs
- Standardized Programming interfaces like JDBC