

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Institut National de Formation en Informatique

(INI) Oued Smar

MEMOIRE

Pour l'obtention du diplôme de

MAGISTER EN INFORMATIQUE

(Option Systèmes d'Informations)

Vérification formelle des propriétés de sécurité des
logiciels

Réalisé par : M^{elle} MOHAND OUSSAID Linda

Devant le jury composé de :

Président : Mme H. DRIAS Professeur à l'USTHB

Rapporteur : Mr M. AHMED NACER Professeur à l'USTHB

Examineur : Mme A. AISSANI Professeur à l'USTHB

Examineur : Mr M. KOUDIL Maître de conférence à l'INI

INI, 2006

Résumé

Cette thèse présente une démarche de vérification formelle des propriétés de sécurité des logiciels. En effet, après étude des approches possibles : preuve formelle et model checking, nous avons identifié le model checking comme étant l'approche la plus adéquate pour notre problématique. Ainsi, la démarche proposée est fondée sur une combinaison de la technique de model checking et des mécanismes : d'annotation et d'abstraction basés sur l'interprétation abstraite.

Pour procéder au model checking, nous avons commencé par exprimer les propriétés de sécurité (confidentialité, intégrité, disponibilité) dans une logique temporelle appropriée. Ensuite, nous avons spécifié les systèmes auxquels pouvait s'appliquer notre démarche à savoir : les systèmes dont l'évolution peut être assimilée à un programme impératif réduit aux structures de contrôle de base.

Moyennant deux étapes : la première qui enrichit le code source par des informations relatives à la propriété à vérifier et la deuxième qui débarrasse le code source enrichi des informations superflues à la vérification de la propriété de sécurité, nous avons transformé le code source d'un programme impératif en un programme abstrait qui exhibe les instructions concernées par les propriétés à vérifier tout en faisant abstraction du reste.

L'annotation aussi bien que l'abstraction ont été développées par interprétation abstraite autour d'une correspondance de Galois qui assure que le programme abstrait correspond au programme initial.

Cette démarche a été illustrée par un exemple qui consiste en un gestionnaire de transactions bancaires simplifié.

Mots clés

Vérification formelle, model checking, propriétés de sécurité, interprétation abstraite.

Abstract

This thesis presents a formal verification of software security properties. After a brief study of the possible approaches : formal proof and model checking, we identified the model checking as being the most adequate approach for our problem. Indeed, the proposed approach is founded on a combination of the model checking technique and mechanisms : of annotation and abstraction based on abstract interpretation.

To proceed to the model checking, we started by expressing the security properties (confidentiality, integrity, availability) in an appropriate temporal logic. Then, we specified the systems to which our approach could be applied : systems whose evolution can be comparable with an imperative program reduced to the basic control structures.

By means of two stages : the first which enriches the source code by information relating to the property to be checked and the second which disencumber the enriched source code of superfluous information to the checking of the security property, we transformed the source code of an imperative program into an abstract program which exhibits instructions concerned with the properties to check while disregarding remainder.

The annotation as well as the abstraction, were developed by abstract interpretation around a Galois connection which ensures that the abstract program corresponds to the initial program.

This approach is illustrated by an example which consists of a simplified banking transactions manager.

Keywords

Formal verification, model checking, security properties, abstract interpretation

TABLE DES MATIERES

Introduction générale.....	1
-----------------------------------	----------

Première partie : Etat de l'art

1 Chapitre 1 : La sécurité des logiciels	3
1.1 Introduction	3
1.2 Définition	3
1.3 Le règlement (politique) de sécurité.....	3
1.3.1 Typologie des règlements de sécurité	4
1.3.1.1 Le modèle de gestion d'accès obligatoire (par mandats) (MAC).....	4
1.3.1.2 Le modèle de gestion d'accès discrétionnaire (DAC)	4
1.3.2 Modélisation formelle des règlements de sécurité	5
1.4 Les propriétés de sécurité	6
1.5 Conclusion.....	7
2 Chapitre 2 : La vérification formelle.....	8
2.1 Introduction	8
2.2 Définition de la vérification formelle.....	8
2.3 Analyse dynamique Vs analyse statique	9
2.4 Les approches possibles de vérification formelle	9
2.4.1 La preuve formelle	9
2.4.2 Le model checking	10
2.5 Le model checking	11
2.5.1 Définition du model checking.....	11
2.5.2 Les types de propriétés	11
2.5.2.1 Les propriétés de sûreté (safety) ou propriétés d'invariance.....	11
2.5.2.2 Les propriétés de vivacité (liveness)	12
2.5.2.3 Les propriétés d'équité (fairness).....	12
2.5.3 Les types de model checking	12
2.5.3.1 Approche basée sur la logique (hétérogène) :	12
2.5.3.2 Approche basée sur le comportement (homogène) :	13
2.5.4 Modélisation du système	13
2.5.4.1 Les structures de kripke.....	13
2.5.4.2 Les diagrammes de décisions binaires (BDDs).....	15

2.5.5	Représentation des propriétés.....	16
2.5.5.1	Les logiques du temps linéaire.....	17
	La logique LTL (Linear Temporal Logic).....	17
	La syntaxe de LTL.....	18
	La sémantique de LTL.....	19
	Principe de l'algorithme de model checking LTL.....	19
	Complexité du model checking LTL.....	20
	Limites du model checking LTL.....	20
	Les extensions de LTL.....	20
	Les outils de vérification pour LTL.....	21
2.5.5.2	Les logiques du temps arborescent.....	21
	La logique CTL (Computational Tree Logic).....	21
	La syntaxe de CTL.....	22
	La sémantique de CTL.....	23
	Principe de l'algorithme de model checking CTL.....	23
	Complexité du model checking CTL.....	26
	Limites du model checking CTL.....	27
	Les extensions de CTL.....	27
	Les outils de vérification pour CTL.....	28
2.5.5.3	Comparaison entre les logiques du temps linéaire et arborescent.....	28
	L'expressivité.....	28
	L'efficacité d'évaluation.....	29
2.6	Conclusion.....	30
3	Chapitre 3 : L'interprétation abstraite.....	31
3.1	Introduction.....	31
3.2	Définition de l'interprétation abstraite.....	31
3.3	Principe de l'interprétation abstraite.....	32
3.4	Quelques définitions.....	32
3.4.1	Modèle de calcul.....	32
3.4.2	Définition de la sémantique d'un programme.....	32
3.4.2.1	Domaine.....	32
3.4.2.2	Ordre Partiel.....	33
3.4.2.3	Ensemble partiellement ordonné et plus petit élément.....	33
3.4.2.4	Borne supérieure.....	33

3.4.2.5	Chaînes	33
3.4.2.6	Fonction continue	34
3.4.2.7	Théorème du point fixe (Tarski, Kleene)	34
3.4.2.8	Treillis complet	34
3.4.3	Propriétés d'une sémantique abstraite	34
3.5	Les types de sémantiques	35
3.5.1	Les sémantiques opérationnelles	35
3.5.2	Les sémantiques dénotationnelles (fonctionnelles)	36
3.5.3	Les sémantiques axiomatiques (sémantiques à états)	36
3.6	Conclusion	36
<u>Deuxième partie : Conception</u>		
4	Chapitre 4 : La démarche proposée	38
4.1	Introduction	38
4.2	Choix du système à vérifier	38
4.3	Démarche proposée	40
4.4	Réécriture des propriétés de sécurité	41
4.5	Annotation et abstraction du programme	43
4.6	Conclusion	46
5	Chapitre 5 : L'annotation du programme	47
5.1	Introduction	47
5.2	Annotation du programme	47
5.3	Sémantique des programmes à vérifier	50
5.4	La fonction d'annotation pour la confidentialité	53
5.5	Sémantique des programmes annotés	55
5.6	La fonction de réduction	58
5.7	La preuve de correction	59
5.8	Conclusion	74
6	Chapitre 6 : L'abstraction du programme	75
6.1	Introduction	75
6.2	Abstraction du programme	75
6.3	La fonction d'abstraction	77
6.4	Sémantique des programmes abstraits	79
6.5	La fonction de concrétisation	81

6.6	La preuve de correction.....	84
6.7	Conclusion.....	99
7	Chapitre 7 Mise en oeuvre.....	100
7.1	Introduction.....	100
7.2	Développement de l'interprète.....	100
7.3	Traduction dans le langage du modèle checker	104
7.4	Application à une étude de cas.....	105
7.5	Conclusion.....	109
	Conclusion générale et perspectives	110
	Références bibliographiques et électroniques.....	112
	Annexe : Syntaxe et sémantique de SMV.....	116