Étude de propriétés d'opacité temporisée à l'aide de vérification temporisée paramétrée*

Dylan Marinho
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy

Résumé

Une fuite d'informations peut avoir des conséquences dramatiques sur la sécurité d'un système. Parmi ces failles, une fuite d'informations temporelle se produit lorsqu'un attaquant peut déduire des informations internes confidentielles en basant son attaque sur une observation temporisée du système. Nous modélisons ici le système par un automate temporisé, et considérons qu'un attaquant a accès (uniquement) au temps d'exécution du système. Nous abordons deux problèmes d'opacité temporisée : déterminer les temps d'exécution pour lesquels le système est sécurisé (c'est-à-dire pour lesquels l'attaquant ne peut pas déduire d'information confidentielle) et déterminer si un système est sécurisé pour tous ses temps d'exécution possibles.

1 Introduction

Étant donné un ensemble de chemins qui révèlent un secret, l'opacité caractérise le fait que s'il existe une exécution du système qui révèle le secret, il existe une autre exécution, avec la même observation, qui ne révèle pas ce secret. Notre étude porte sur une forme d'opacité dans laquelle l'observation se fait uniquement sur le temps nécessaire pour atteindre une localité finale.

2 Problèmes étudiés

Dans ce travail, un système est modélisé par un automate temporisé paramétré (*Parametric Timed Automaton (PTA)*) [AHV93]. Un PTA $\mathcal{A}$ est un automate fini, composé d'un ensemble de localités et d'un ensemble d'actions, que l'on étend avec : *i*) un ensemble d'horloges $\mathbb{X} = \{x, y, \ldots\}$, qui sont des variables réelles augmentant linéairement ; *ii*) un ensemble de paramètres $\mathbb{P} = \{p_1, p_2, \ldots\}$. On définit ensuite : *i*) des invariants sur les localités (propriétés à vérifier, sous la forme d'une

*Ce travail est tiré d'un article récemment accepté à ACM TOSEM [And+22], en collaboration avec Étienne André, Didier Lime et Jun Sun. Ce travail est soutenu par le projet ANR-NRF ProMiS (ANR-19-CE25-0015).

(a) Un PTA $\mathcal{A}$ (b) Un TA $v(\mathcal{A})$, où $v(p) = 3$

FIGURE 1 – Un exemple de PTA et de TA

contrainte linéaire, pour rester dans une localité) ; *ii*) des gardes sur les transitions (propriétés à vérifier pour activer une transition) ; *iii*) la réinitialisation d'horloges lors d'une transition. Un exemple de PTA est donné par la figure 1a.

Si l'on considère un PTA $\mathcal{A}$ et une fonction d'évaluation $v : \mathbb{P} \to \mathbb{Q}$, on définit l'automate temporisé $v(\mathcal{A})$ (*Timed automaton – TA*) [AD94] comme étant un PTA dans lequel tous les paramètres sont évalués en fonction de v. Un exemple de TA est donné par la figure 1b.

Un PTA est L/U [Hun+02] si l'ensemble de ses paramètres est partitionné en deux sous-ensembles : celui des paramètres toujours comparés en borne inférieure dans les gardes et invariants, et celui des paramètres toujours comparés en borne supérieure. Le PTA figure 1a est L/U : p est toujours comparé en borne supérieure.

2.1 Opacité temporisée

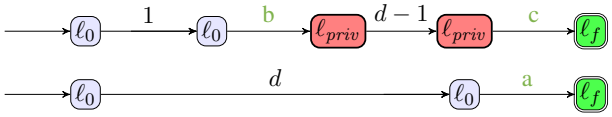
On considère que l'attaquant a accès seulement au temps total d'exécution. Le secret est formalisé par la visite, ou non, d'une localité privée ℓ_{priv} sur le chemin conduisant à la localité finale ℓ_f. On cherche donc à définir qu'un système est sécurisé si l'attaquant ne peut pas déterminer si la localité privée a été visitée.

Définition 1 (Opacité temporisée)**.** Un système est *opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f* pour un temps d s'il existe deux chemins de ℓ_0 à ℓ_f de temps d *i*) l'un passant par ℓ_{priv} et *ii*) l'autre ne passant pas par ℓ_{priv}.

Le système est dit *complètement opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f* s'il est opaque pour tous ses temps d'exécution possibles.

Exemple 1. On considère le TA donné en figure 1b.

— Il existe deux exécutions de durée d pour tout $d \in [2, 3]$:



Le système est donc *opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f* pour toute durée dans $[2, 3]$;

— Mais il est possible d'atteindre ℓ_f avec une exécution de durée $1,5$ ne passant pas par ℓ_{priv}, sans que cela ne soit possible en passant par ℓ_{priv} : le système n'est donc pas complètement opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f.

2.2 Opacité sur les automates temporisés

Un TA étant défini uniquement avec des horloges et des constantes, on s'intéresse au fait *i*) de pouvoir déterminer quels sont les temps d'exécution sécurisés (TOCP) et *ii*) de pouvoir déterminer si le modèle est sécurisé pour tous ses temps d'exécution (FTODP). Les deux problèmes suivants considèrent en entrée un TA $v(\mathcal{A})$, une localité privée ℓ_{priv} et une localité finale ℓ_f.

Calcul d'opacité temporisée (***Timed Opacity Computation Problem – TOCP***)**:**
Calculer les temps d'exécution D pour lesquels $v(\mathcal{A})$ est opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f.

Problème de décidabilité de l'opacité temporisée complète (***Full Timed Opacity Decision Problem – FTODP***)**:**
$v(\mathcal{A})$ est-il complètement opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f ?

2.3 Opacité sur les automates temporisés paramétrés

On définit également des problèmes analogues sur les PTA, dans lesquels l'objectif est de déterminer s'il existe des fonctions d'évaluations qui permettent de vérifier la propriété. Ces problèmes considèrent en entrée un PTA $\mathcal{A}$, une localité privée ℓ_{priv} et une localité finale ℓ_f.

Vide de l'opacité (***Timed Opacity Emptiness Problem – TOEP***)**:**
L'ensemble des fonctions d'évaluations v telles que $v(\mathcal{A})$ est opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f pour un ensemble non vide de temps d'exécution est-il vide ?

Vide de l'opacité complète (***Full Timed Opacity Emptiness Problem – FTOEP***)**:**
L'ensemble des fonctions d'évaluations v telles que $v(\mathcal{A})$ est complètement opaque vis-à-vis de ℓ_{priv} sur le chemin vers ℓ_f est-il vide ?

On définit également de façon analogue les problèmes de synthèse de paramètres, où l'on cherche à synthétiser toutes les fonctions d'évaluations v pour lesquelles les propriétés susmentionnées sont vérifiées.

3 Résultats théoriques

Nous avons étudié la décidabilité (ou la calculabilité, suivant le problème étudié) de chacun des problèmes présentés dans la section 2. Les différents résultats sont présentés dans le tableau 1.

Nous observons alors que les problèmes définis sur des TA restent décidables, mais deviennent presque tous indécidables dès lors que nous considérons des paramètres. Le seul cas d'étude qui permet de conserver la décidabilité est l'opacité temporisée appliquée aux L/U-PTA. Cependant, nous avons montré que la synthèse est infaisable en pratique. Plus précisément, nous montrons qu'il est impossible de représenter l'ensemble des solutions à l'aide d'un formalisme pour lequel le vide de

TABLEAU 1 – Récapitulatif des résultats théoriques obtenus

Problème		Opacité temporisée	Opacité temporisée complète
Calculabilité (TOCP) Décidabilité (FTODP)	TA	calculable [**And+22**, Prop. 5.2]	décidable [**And+22**, Prop. 5.3]
Vide ((F)TOEP)	PTA	indécidable [**And+22**, Thm. 6.1]	indécidable [**And+22**, Thm. 7.2]
	L/U-PTA	décidable [**And+22**, Thm. 6.2]	indécidable [**And+22**, Thm. 7.4]
Synthèse	L/U-PTA	infaisable [**And+22**, Prop. 6.4]	*de facto infaisable*

l'intersection est décidable : cela exclut donc l'utilisation des unions de polyèdres, un formalisme pourtant souvent utilisé pour la synthèse de paramètres.

4 Expérimentations

Nous avons défini une procédure permettant de déterminer les fonctions d'évaluations qui rendent un PTA opaque. En raison de l'indécidabilité du problème, cette procédure n'a pas de garantie de terminaison. Nous avons ensuite utilisé IMITATOR [And21] (un logiciel de vérification temporisée paramétrée prenant en entrée une extension des PTA) afin de donner une preuve de concept de notre méthode.

Notre approche a été appliquée à des modèles de la littérature [AMP21] ainsi qu'à des programmes Java manuellement traduits en PTA [STAC].

Références

[AD94] Rajeev ALUR et David L. DILL. "A theory of timed automata". In : *Theoretical Computer Science* 126.2 (avr. 1994), p. 183-235. ISSN : 0304-3975. DOI : 10.1016/0304-3975(94)90010-8.

[AHV93] Rajeev ALUR, Thomas A. HENZINGER et Moshe Y. VARDI. "Parametric real-time reasoning". In : *STOC*. San Diego, California, United States : ACM, 1993, p. 592-601. DOI : 10.1145/167088.167242.

[AMP21] Étienne ANDRÉ, Dylan MARINHO et Jaco van de POL. "A Benchmarks Library for Extended Parametric Timed Automata". In : *TAP 2021*. 2021. DOI : 10.1007/978-3-030-79379-1_3.

[And+22] Étienne ANDRÉ, Didier LIME, Dylan MARINHO et Jun SUN. "Guaranteeing Timed Opacity using Parametric Timed Model Checking". In : *TOSEM* (2022).

[And21] Étienne ANDRÉ. "IMITATOR 3 : Synthesis of Timing Parameters Beyond Decidability". In : *CAV 2021*. 2021. DOI : 10.1007/978-3-030-81685-8_26.

[Hun+02] Thomas HUNE, Judi ROMIJN, Mariëlle STOELINGA et Frits W. VAANDRAGER. "Linear parametric model checking of timed automata". In : *JLAP* 52-53 (2002). DOI : 10.1016/S1567-8326(02)00037-1.

[STAC] STAC. URL : https://github.com/Apogee-Research/STAC/.