Non-interference temporisée avec observation partielle et mémoire bornée

Anthony Spriet, Didier Lime, Olivier-H Roux

Non-interférence temporisée avec observation partielle et mémoire bornée*

Anthony Spriet, Didier Lime, and Olivier H. Roux

Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes
{anthony.spriet,didier.lime,olivier-h.roux}@ec-nantes.fr

Abstract

Nous étudions une propriété de non-interférence temporisée pour les systèmes de sécurité modélisés sous forme d'automates temporisés, dans lesquels un utilisateur de bas niveau de sécurité ne doit pas être en mesure de déduire l'occurrence d'une action d'un niveau de sécurité élevé. Nous supposons deux modèles d'attaque différents : dans le premier, l'utilisateur malveillant (de bas niveau) a la capacité d'observer totalement l'état du système mais pas de le mémoriser ; dans le deuxième, il ne peut qu'observer partiellement l'état du système mais peut le mémoriser au cours de son exécution.

Nous commençons par présenter une propriété de non-interférence existant dans la littérature garantissant la sécurité du système vis-à-vis du premier modèle d'attaque. Nous montrons que les preuves de décidabilité de cette propriété sont à ce jour incomplètes et en fournissons une nouvelle. Nous formalisons ensuite une propriété de non-interférence qui garantit la sécurité du système vis-à-vis du second modèle d'attaque et nous prouvons l'indécidabilité de cette propriété lorsque l'attaquant peut avoir une mémoire arbitrairement grande, c'est-à-dire lorsqu'il est capable de mémoriser des séquences d'observations, ainsi que la durée entre deux observations successives, de n'importe quelle longueur. Nous supposons ensuite une mémoire limitée pour l'attaquant et montrons que la propriété peut alors être décidée. Finalement, nous montrons que la propriété peut être décidée avec une complexité PSPACE pour une sous-classe d'automates temporisés assurant une durée finie entre deux observations distinctes mémorisées par l'attaquant.

Keywords: Non-interférence, Automates temporisés, Observation partielle.

1 Introduction

La fuite d'informations est un type de vulnérabilité logicielle dans lequel des informations sont involontairement accessibles à des utilisateurs non autorisés.

Diverses propriétés de flux d'informations ont été définies dans la littérature, telles que l'anonymat, la non-interférence, le secret, la confidentialité, l'opacité et la non-déductibilité. Ces propriétés se recoupent et des comparaisons formelles ont été effectuées dans [9] entre opacité, anonymat, non-interférence (sur les traces) et non-déductibilité. Nous nous concentrons ici sur les propriétés de non-interférence qui pourraient aussi être écrites comme de l'opacité.

Un système est dit non interférent [14] si les informations disponibles sur l'interface du système ne sont pas suffisantes pour déduire certaines informations internes classifiées. En d'autres termes, les informations disponibles dans le canal de bas niveau (non classifiées, observables par la plupart des utilisateurs) ne contiennent aucun indice sur les informations uniquement disponibles dans le canal de haut niveau (informations classifiées, accessibles uniquement à certains utilisateurs ou à aucun utilisateur). Une autre façon de le dire est

*Ce travail a été partiellement financé par le projet ANR ProMiS ANR-19-CE25-0015.

qu'une séquence d'entrées de bas niveau produira toujours les mêmes sorties de bas niveau, indépendamment de ce qui se passe sur les entrées de haut niveau.

Il a été montré qu'un système peut être non interférent lorsqu'on considère des séquences discrètes, mais interférent lorsque les dates de chaque observation des différentes entrées et sorties sont mesurées et mémorisées par un observateur malveillant [15, 11, 8, 16].
Dans [6], Barbuti *et al* ont proposé deux notions de non-interférence temporelle dans les systèmes modélisés par des automates temporisés. La première est une notion basée sur les traces, dans laquelle l'attaquant peut observer les actions consécutives effectuées par le système et tente de deviner si une action cachée s'est produite. Cette notion a depuis été affinée en considérant des relations de simulation ou en se basant sur des algorithmes de synthèse de contrôleur [12, 7, 13]. La seconde approche est basée sur les états : l'attaquant peut observer directement l'état du système mais ne voit aucune action. Si l'état observé ne peut pas être atteint en utilisant uniquement des actions de bas niveau, alors l'attaquant est en mesure de déduire l'utilisation d'une action de haut niveau. Cette approche basée sur les états a également été affinée depuis en étant étendue aux automates temporisés paramétriques [3]. D'autres notions de non-interférence ont également été proposées dont le cadre commun consiste à garantir qu'aucune information n'est divulguée pendant le temps d'exécution d'un automate temporisé avec des emplacements finaux [4, 18, 2].

Dans l'arcticle [17] nous avons introduit une nouvelle propriété de non-interférence basée sur les états (POS-NNI) et avons discuté de sa pertinence lorsqu'il s'agit de modéliser des attaquants réalistes. Nous y avons montré que le problème de vérification de la POS-NNI est indécidable lorsque la mémoire de l'attaquant est infinie et avons montré que sa vérification est PSPACE-complète dans le cas d'un attaquant à mémoire bornée pour une classe d'automates.

Ces travaux font suite à l'article [17]. Les preuves des résultats déjà présents dans cette publication ont donc été omises.

Contributions Nous approfondissons l'étude de la POS-NNI et montrons qu'elle est décidable dans le cas d'un attaquant à mémoire bornée dans le cas général des automates temporisés. Nous montrons également que la propriété St-NNI issue de la littérature [12] est également décidable dans le cas général après avoir montré que la preuve d'origine ne concernait en réalité que les automates temporisés à horloges bornées.

Dans la Section 2, on rappelle les définitions utilisées pour formaliser les propriétés de non-interférences temporisées et présentons les constructions appliquées aux automates temporisés afin de modéliser les différents accès à l'information. Dans la Section 3, on évoque un critère de non-interférence issue de la littérature et montrons que les preuves de décidabilités disponibles sont incomplètes et en fournissons une preuve définitive. Dans la Section 4, on introduit une nouvelle propriété de non-interférence, la POS-NNI, et démontrons sont indécidabilité dans le cas d'un attaquant disposant d'une mémoire infinie. Enfin, dans la Section 5, nous fournissons une preuve de la décidabilité de la POS-NNI lorsque l'attaquant a une mémoire bornée. De plus, nous mettons en évidence une classe d'automates pour laquelle la vérification de la POS-NNI à mémoire bornée est PSPACE-complète.

2 Définitions

2.1 Les automates temporisés

Les automates temporisés [1] sont un des formalismes utilisés pour modéliser les systèmes temps réel. Ils consistent en l'ajout d'horloges à valeurs réelles aux automates finis afin d'y

représenter fidèlement les comportements temporisés. Par la suite nous nous concentrerons sur les *automates de sécurités*. Il s'agit d'automates temporisés qui voient leur ensemble de transitions partitionné en deux sous-ensembles, chacun modélisant un utilisateur avec un niveau d'accès différent à l'information et à la commande du système.

Définition 1 (Horloges et valuation)**.** *Une horloge est une variable à valeur réelle qui évolue avec l'écoulement du temps.*

Soit un ensemble $\mathbb{X} = \{x_1, x_2, ..., x_H\}$ *d'horloges.*

Une valuation est une fonction $\nu : \mathbb{X} \to \mathbb{R}_{\geq 0}$*. L'ensemble des valuations de* $\mathbb{X}$ *est noté* $\mathbb{R}^{\mathbb{X}}_{\geq 0}$*,* $d \in \mathbb{R}_{\geq 0}, \nu + d$ *désigne la valuation telle que* $(\nu + d)(x) = \nu(x) + d$*.*

Soit $R \subseteq \mathbb{X}$*, nous définissons la réinitialisation d'une évaluation* ν*, notée* $[\nu]_R$ *telle que* $[\nu]_R(x) = 0$ *si* $x \in R$*, et* $[\nu]_R(x) = \nu(x)$ *sinon.*

Définition 2 (Contrainte temporelle)**.** *Une contrainte temporelle* g *contraignant* $\mathbb{X}$ *est définie par un système d'inéquations de la forme* $x \bowtie d$ *avec* $d \in \mathbb{N}$ *et* $\bowtie \in \{\leq, <, >, \geq\}$*.*

On note $\nu \models g$ *si et seulement si le système d'inéquations représenté par* g *est satisfait lorsque chaque horloge* x *est remplacée par sa valuation* $\nu(x)$*.*

Définition 3 (ϵ**-Automate Temporisé (**ϵ**-TA)).** *Un* ϵ*-TA* $\mathcal{A}$ *est un tuple* $\mathcal{A} = (\Sigma \cup \{\epsilon\}, L, l_0, \mathbb{X}, I, E)$ *où :* Σ *est un ensemble fini d'actions,* ϵ *est l'unique "action silencieuse",* L *est un ensemble fini de localités,* l_0 *est la localité initiale contenue dans* L*,* $\mathbb{X}$ *est un vecteur d'horloges,* I *attribue à chaque localité* $l \in L$ *une contrainte temporelle* $I(l)$ *appelée* invariant *de* l*,* E *est un ensemble fini de transitions de la forme* $e = (l, g, a, R, l')$ *avec* $l, l' \in L$ *respectivement les localités source et cible de la transition,* $a \in \Sigma \cup \{\epsilon\}, R \subseteq \mathbb{X}$ *l'ensemble d'horloges étant réinitialiser par* e *et* g *une contrainte temporelle appelée "garde" de* e*.*

Définition 4 (**Sémantique des automates temporisés**)**.** *Soit un* ϵ*-TA* $\mathcal{A} = (\Sigma \cup \{\epsilon\}, L, l_0, \mathbb{X}, I, E)$*, la sémantique de* $\mathcal{A}$ *est donné par le système de transition* $(S, s_0, \to)$*, pour lequel :*

- $S = \{(l, \nu) \in L \times \mathbb{R}^{\mathbb{X}}_{\geq 0} \mid \nu \models I(l)\}$
- $s_0 = (l_0, \boldsymbol{0})$ *où* $\boldsymbol{0}$ *est la valuation de* $\mathbb{X}$ *telle que toutes les horloges sont à 0.*
- $\to$ *est l'ensemble le plus grand de transitions discrètes et continues de la forme :*
  - *Transitions discrètes :* $(l, \nu) \xrightarrow{a} (l', \nu')$ *avec* $(l, \nu), (l', \nu') \in S$ *telle qu'il existe* $e = (l, g, a, R, l') \in E$ *avec* $\nu' = [\nu]_R$ *et* $\nu \models g$
  - *Transitions continues :* $(l, \nu) \xrightarrow{d} (l, \nu + d)$ *avec* $d \in \mathbb{R}_{\geq 0}$ *et* $\forall d' \in [0, d], (l, \nu + d') \in S$

Définition 5 (**Exécution**)**.** *Soit* $\mathcal{A} = (\Sigma \cup \{\epsilon\}, L, l_0, \mathbb{X}, I, E)$ *un* ϵ*-TA et* $(S, s_0, \to)$ *sa sémantique. Une exécution (finie)* ρ *de* $\mathcal{A}$ *est une séquence de la forme :* $\rho = s_0 e_0 s_1 e_1 \cdots s_n$ *avec* $\forall i, s_i \in S$ *et* $(s_i, e_i, s_{i+1}) \in \to$*, qu'on notera aussi* $\rho = s_0 \xrightarrow{e_0} s_1 \xrightarrow{e_1} \cdots \xrightarrow{e_{n-1}} s_n$*.*

Une exécution ρ *peut toujours être représenté sous la forme :*$\rho = s_0 \xrightarrow{d_1} s_1 \xrightarrow{a_2}\xrightarrow{d_2} s_2 \xrightarrow{a_2} ...s_{n-1} \xrightarrow{a_n}\xrightarrow{d_n} s_n$ *où les transitions discrètes et continues s'alternent (les noms des états ont été omis).*

On note $\Psi(\mathcal{A})$ l'ensemble des exécutions de $\mathcal{A}$. On dit d'un état s qu'il est atteignable dans $\mathcal{A}$ s'il existe une exécution $\rho \in \Psi(\mathcal{A})$ telle que $s \in \rho$. On note $Q^{\mathcal{A}}$ l'ensemble des états atteignable de $\mathcal{A}$.

2.2 Langage Temporisé

Les séquences d'actions décrites par les exécutions d'un automate, associées à leurs dates, décrivent le comportement de l'automate.

Définition 6 (Mot Temporisé et Trace)**.** *Un mot temporisé défini sur l'alphabet* Σ *est une séquence finie* $w = (a_0, t_0)(a_1, t_1)...(a_n, t_n)$ *telle que* $\forall i \geq 0, a_i \in \Sigma, t_i \in \mathbb{R}_{\geq 0}$.

Soit une exécution $\rho = s_0 \xrightarrow{d_1} s_1 \xrightarrow{a_2}\xrightarrow{d_2} s_2 \xrightarrow{a_2} ...s_{n-1} \xrightarrow{a_n}\xrightarrow{d_n} s_n$, *sa* trace *est le mot temporisé* $(a_1, d_1)(a_2, d_1 + d_2) \cdots (a_n, \sum_{i=1}^{n-1} d_i)$

On note $\mathcal{U}(\Sigma)$ l'ensemble des mots temporisés définis sur l'alphabet Σ.

Définition 7 (**Langage Temporisé**)**.** *Soit un* ϵ*-TA* $\mathcal{A}$*, son langage temporisé* $\mathcal{L}(\mathcal{A})$ *est l'ensemble des traces de l'ensemble des ses exécutions.*

2.3 Automates de Sécurité Temporisés

Afin de modéliser différents niveau d'accès à l'information dans les systèmes modélisés par des automates, nous définissons des sous-ensembles d'actions correspondant aux différents utilisateurs. Les actions de "haut niveau" ne sont utilisable que par des utilisateurs privilégiés tandis que les actions de "bas niveau" sont accessibles à tous.

Définition 8 (**Automate de Sécurité Temporisé**)**.** *Un automate de sécurité temporisé* $\mathcal{A} = (\Sigma \cup \{\epsilon\}, L, l_0, \mathbb{X}, I, E)$ *est un* ϵ*-TA dont l'ensemble des actions* Σ *est partitionné en deux sous-ensembles* Σ_{bas} *et* Σ_{haut} *tels que* $\Sigma_{\mathsf{bas}} \cap \Sigma_{\mathsf{haut}} = \emptyset$ *et* $\Sigma_{\mathsf{bas}} \cup \Sigma_{\mathsf{haut}} = \Sigma$.

Afin de comparer les comportements de l'automate pour ses différents utilisateur, on définit un automate restreint aux actions de bas niveau. Fig. 1 donne un exemple de TA de sécurité $\mathcal{A}$ et sa restriction $\mathcal{A}_{\setminus \Sigma_{\mathsf{haut}}}$.

Définition 9 (**Automate Temporisé Restreint**)**.** *Soit* $\mathcal{A} = (\Sigma, L, l_0, \mathbb{X}, I, E)$ *un TA et* $\Gamma \subseteq \Sigma$. *On définit le TA* Γ*-restreint* $\mathcal{A}_{\setminus \Gamma} = (\Sigma \setminus \Gamma, L, l_0, \mathbb{X}, I, E_{\setminus \Gamma})$ *telle que* $(l, g, a, R, l') \in E_{\setminus \Gamma}$ *si et seulement si* $a \in (\Sigma \setminus \Gamma)$ *et* $(l, g, a, R, l') \in E$.

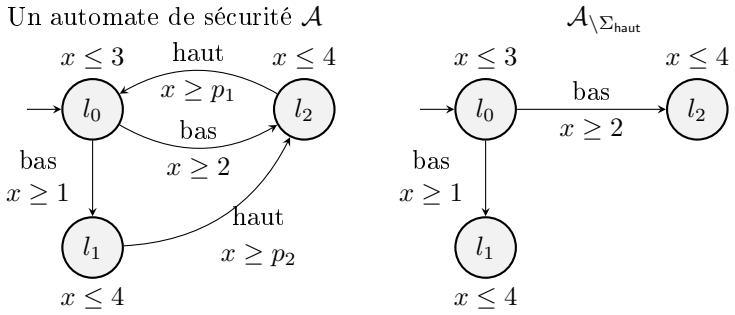

Figure 1: Un automate de sécurité $\mathcal{A}$ et $\mathcal{A}_{\setminus \Sigma_{\mathsf{haut}}}$

3 Propriétés de non-interférence basées sur l'état dans la littérature

A notre connaissance, la première proposition d'une propriété de non-interférence basées sur l'état dans les automates temporisés nous vient de Barbuti et Tesei [6]. Soit un automate de sécurité $\mathcal{A}$ et une période entière n, un automate d'attaque $\texttt{Interf}^n_{\Sigma_{\text{haut}}}$ est défini figure 2.

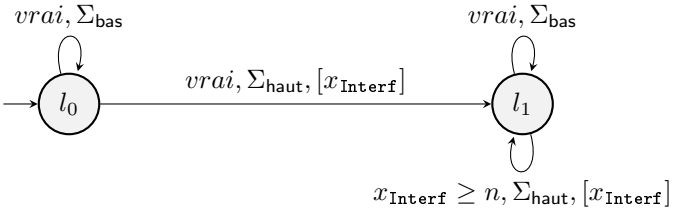

Figure 2: Automate d'attaque $\texttt{Interf}^n_{\Sigma_{\text{haut}}}$.

L'automate "attaqué" est défini par le *produit synchronisé* [1] $\mathcal{B} = \mathcal{A}||\texttt{Interf}^n_{\Sigma_{\text{haut}}}$. la propriété de non-interférence appelée *n-state-non-interférence* est alors satisfaite si et seulement si :

$$Q^{\mathcal{A}_{\setminus \Sigma_{\text{haut}}}} = Q^{\mathcal{B}}$$

Une manière de voir cette propriété est de considérer le sous-ensemble des états atteignables par les actions de bas niveau uniquement comme étant les "comportements normaux" du système et les autres états comme étant les "comportements anormaux". Si l'ensemble d'états "anormaux" est vide cela signifie que l'attaquant ne parvient pas à déduire d'informations à partir de l'état tant que les actions de haut niveau adviennent à une fréquence inférieure ou égale à $\frac{1}{n}$.

Pour définir une propriété de non-interférence basée sur l'état on peut aussi se représenter un attaquant sans mémoire capable d'observer à tout instant l'état du système (localité et valuation d'horloge) tentant de deviner si une action de haut niveau s'est produite. Parce qu'il n'a pas de mémoire, cet attaquant est limité à deviner à partir d'une unique observation de l'état du système, si l'état est inatteignable sans action de haut niveau. Ce cadre correspond à un critère baptisé *St-NNI* (State non-interference) par Gardey *et al* [12]. Un automate de sécurité satisfait la St-NNI si est seulement si :

$$Q^{\mathcal{A}} = Q^{\mathcal{A}\setminus \Sigma_{\text{haut}}}$$

Ce qui est en réalité équivalent à une 0-state-non-interference telle que définie par Barbuti *et al*.

Dans les deux cas, les preuves de décidabilité des propriétés données par leurs auteurs font simplement référence à la décidabilité de l'atteignabilité d'un état dans les automates temporisés. Or, décider l'atteignabilité d'un état en particulier n'implique pas directement la possibilité de calculer l'ensemble des états atteignables. Plus spécifiquement, la preuve de décidabilité de l'atteignabilité d'un état repose sur le construction d'un graphe des régions. Si cette abstraction préserve bien l'atteignabilité des localités, elle ne préserve pas fidèlement les valuations d'horloges car les régions sont définies à l'aide d'une constante au delà de laquelle une grande variété d'états est regroupée au sein d'une même région, ceci dans le but de produire un graphe fini. Pour illustrer ce point, nous donnons sur la figure 3 un automate de sécurité

défini avec deux horloges (x et y) pour lequel l'automate restreint aux actions de bas niveau est représenté par le même graphe des régions alors que ses états atteignables diffèrent de l'automate d'origine :

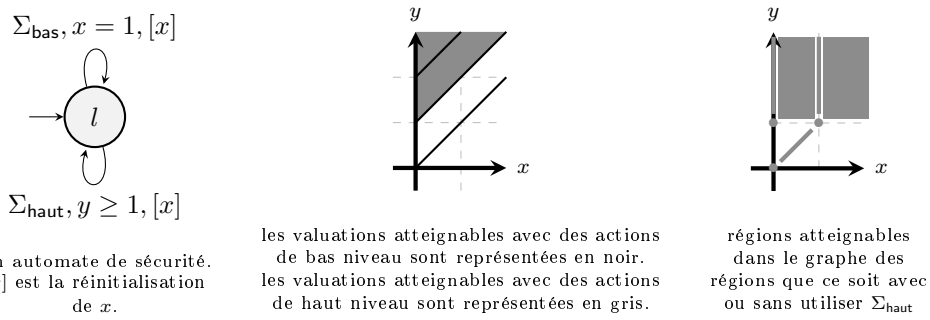

Figure 3: Exemple d'un automate de sécurité pour lequel le graphe des régions ne dépend pas du niveau de l'utilisateur.

Cet exemple est minimal, mais il est évidement possible de définir des automates plus complexes afin d'assurer que les états interférents soient plus éloignés encore de la constante généralement utilisée pour définir les régions (la plus grande constante utilisée dans la syntaxe de l'automate, ici 1). Bien que le graphe des régions ne permette pas de décider la St-NNI de manière directe, il existe une représentation complète de l'ensemble des états atteignables d'un automate. Dans leurs travaux, *Comon et Jurski*[10] montrent que dans l'arithmétique $\mathcal{R}$ définie par les formules du premier ordre sur la structure $(\mathbb{R}, \mathbb{Z}(.), 0, 1, +, \leq)$, où $\mathbb{R}$ sont les réels et $\mathbb{Z}(.)$ est la fonction partie entière; on peut prendre deux localités l et l' d'un automate $\mathcal{A}$, et produire une formule $\phi^{\mathcal{A}}_{l,l'}(v, v')$ de $\mathcal{R}$ telle que $\phi^{\mathcal{A}}_{l,l'}(v, v')$ est vraie si et seulement si il existe une exécution de $\mathcal{A}$ allant de (l, v) à (l, v). L'arithmétique mixte $\mathcal{R}$ a été démontrée décidable et ayant la même complexité que l'arithmétique de Presburger par *Weispfenning* ([19]), c'est à dire 2-EXPTIME.

Dans ce qui suit, nous utilisons ces résultats pour décider la St-NNI. Cependant le résultat de *Comon et Jurski* a été démontré dans le cadre d'automates définis sans invariants sur les localités. Nous commençons donc par étendre leur résultats à notre définition des automates.

Lemme 1. *Soit deux localités l et l' d'un automate $\mathcal{A}$ défini avec des invariants sur les localités, on peut produire une formule $\phi^{\mathcal{A}}_{l,l'}(v, v')$ dans $\mathcal{R}$ telle que $\phi^{\mathcal{A}}_{l,l'}(v, v')$ est vraie si et seulement si il existe une exécution de $\mathcal{A}$ allant de (l, v) à (l', v').*

Preuve. Soit un automate temporisé $\mathcal{A} = (\Sigma, L, l_0, X, I, E)$ défini avec des invariants sur les localités, pour tout $R \subset X$ on définit la fonction $I_R(l)$ s'évaluant à vrai si et seulement si $\exists \nu \in \mathbb{R}^X_{\geq 0}$ telle que $[\nu]_R \models I(l)$, et la fonction $I_{\overline{R}}(l)$ comme $I(l)$ projeté sur l'ensemble $X \backslash R$. Nous définissons ensuite l'automate temporisé sans invariant $\mathcal{B} = (\Sigma, L, l_0, X, E')$ avec $E' = \{(l_s, g \wedge I(l_s) \wedge I_{\overline{R}}(l_t) \wedge I_R(l_t), a, R, l_t), \forall (l_s, g, a, R, l_t) \in E\}$. La formule $\phi^{\mathcal{B}}_{l_0,l}(v_0, v) \wedge v \models I(l)$ est alors vraie dans $\mathcal{R}$ si et seulement si (l, v) est atteignable depuis (l_0, v_0) dans une exécution de $\mathcal{A}$. □

Théorème 1. *La St-NNI est décidable.*

Preuve. Soit un automate de sécurité $\mathcal{A} = (\Sigma_{\mathsf{haut}} \cup \Sigma_{\mathsf{bas}}, L, l_0, X, I, E)$, on montre que $\mathcal{A}$ vérifie la St-NNI si est seulement si la formule $F = \forall l \forall v (\phi^{\mathcal{A}}_{l_0,l}(\mathbf{0}, v) \implies \phi^{\mathcal{A}_{\setminus \Sigma_{\mathsf{haut}}}}_{l_0,l}(\mathbf{0}, v))$ de $\mathcal{R}$ est vraie.

D'abord on remarque que $\phi^{\mathcal{A}}_{l_0,l}(\mathbf{0},v)$ (resp $\phi^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}}_{l_0,l}(\mathbf{0},v)$) est vraie si et seulement si $(l,v) \in Q^{\mathcal{A}}$ (resp $(l,v) \in Q^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}}$). Par conséquent, $F \iff \forall s, s \in Q^{\mathcal{A}} \implies s \in Q^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}}$ ce qui est équivalent à $Q^{\mathcal{A}} \subseteq Q^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}}$. Puisque pour tout automate de sécurité on $\mathcal{A}$ on a $\Psi(\mathcal{A}^{\setminus\Sigma_{\text{haut}}}) \subseteq \Psi(\mathcal{A})$, la relation $Q^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}} \subseteq Q^{\mathcal{A}}$ est toujours vérifiée. On a donc : F est vraie $\iff Q^{\mathcal{A}} = Q^{\mathcal{A}_{\setminus\Sigma_{\text{haut}}}}$. Par le lemme 1, la formule F peut être produite pour tout automate de sécurité $\mathcal{A}$ et peut être vérifiée, ce qui conclut la preuve. □

4 Un modèle d'attaque plus réaliste

Comme décrit plus haut, la St-NNI peut être vue comme un critère découlant d'un modèle d'attaque ayant un grand pouvoir d'observation, dans le sens où il est capable d'observer totalement l'état du système, mais avec des capacités de déduction minimales en raison de son absence de mémoire. Si on modélisait un attaquant ayant le même pouvoir d'observation tout en le dotant d'une mémoire, dans le sens où il aurait accès à toutes ses observations passées pour formuler ses déductions, nous modéliserions une exigence de sécurité déraisonnablement élevée. Nous proposons donc un modèle d'attaque ayant une mémoire et étant capable de continuellement observer le système, mais n'étant capable d'observer qu'une partie de l'information. On considère que l'attaquant n'est pas capable d'observer les horloges du système et nous incluons la possibilités que certaines localités soient indistinguables du point de vue de l'attaquant. Du point de vue formel : une fonction d'observation attribue une valeur à chaque localité, l'attaquant est alors capable de mémoriser la valeur qu'il observe ainsi que la date à partir de laquelle il a commencé à l'observer.

Définition 10 (**Fonction d'observation**). *Soit Ω un ensemble fini tel que $|\Omega| \leq |L|$. On appelle* observations *les éléments de Ω. Une* fonction d'observation *est une fonction $\mathcal{O} : L \to \Omega$. Si $\mathcal{O}(l) = \mathcal{O}(l')$ (resp: $\mathcal{O}(l) \neq \mathcal{O}(l')$) on dit de l et l' qu'elles sont indistinguables (resp: distinguable).*

On utilise alors les observations pour définir les exécutions du point de vue de l'attaquant.

Définition 11 (**Exécutions observées et événements observés**). *Soit une exécution $\rho = (l_0, \mathbf{0}) \xrightarrow{d_1} (l_1,\nu_1) \xrightarrow{a_2}\xrightarrow{d_2} (l_2,\nu_2)\dots(l_{n-1},\nu_{n-1}) \xrightarrow{a_n}\xrightarrow{d_n} (l_n,\nu_n)$ et une fonction d'observation $\mathcal{O}$. Nous définissons l'exécution observée $\mathcal{O}(\rho)$ telle que :*

$\mathcal{O}(\rho) = (\mathcal{O}(l_{i_0}),0)(\mathcal{O}(l_{i_1}),\sum_{k=1}^{i_1-1} d_k)\dots(\mathcal{O}(l_{i_{m-1}}),\sum_{k=1}^{i_m-1} d_k)(\mathcal{O}(l_{i_m}),\sum_{k=1}^{i_m} d_k)$, où la séquence $(i_0,i_1,...,i_m)$ est définie par :

- $i_0 = 0$
- *Pour $1 \leq k < m$, i_k est le plus petit indice $j > i_{k-1}$ tel que $\mathcal{O}(l_{i_{k-1}}) \neq \mathcal{O}(l_j)$*
- $i_m = n$

On note $\mathcal{O}(\mathcal{A})$ l'ensemble des exécutions observées de $\mathcal{A}$.

*Quand une exécution observée $\mathcal{O}(\rho)$ contient un élément (a,t) on dit qu'un **événement observé** menant à l'observation a se produit à l'instant t de l'exécution.*

Dit simplement, du fait de l'observation continue du système par l'attaquant, celui ci est capable de mémoriser la date absolue de chaque événement (lorsqu'une transition discrète entre deux localités distinguables se produit). Le dernier élément de l'exécution observée permet de tenir compte du temps écoulé depuis le dernier événement observé, on a donc par construction $\mathcal{O}(l_{i_m}) = \mathcal{O}(l_{i_{m-1}})$.

Exemple 1 (Une exécution et son exécution observée associée)**.** *Soit une exécution* $\rho = (l_0, \boldsymbol{0}) \xrightarrow{d_1,a_1} (l_1,\nu_1) \xrightarrow{d_2,a_2} (l_2,\nu_2) \xrightarrow{d_3,a_3} (l_3,\nu_3) \xrightarrow{d_4,a_4} (l_4,\nu_4) \xrightarrow{d_5,a_5} (l_5,\nu_5) \xrightarrow{d_6} (l_5,\nu_6)$ *avec* $\mathcal{O}(l_0) = \mathcal{O}(l_1) = \mathcal{O}(l_3) = \mathcal{O}(l_4) = o_1, \mathcal{O}(l_2) = o_2$ *et* $\mathcal{O}(l_5) = o_3$*. On a* $\mathcal{O}(\rho) = (o_1,0)(o_2,d_1+d_2)(o_1,d_1+d_2+d_3)(o_3,\sum_{k=1}^{5} d_k)(o_3,\sum_{k=1}^{6} d_k)$

Les événements observés se produisent aux instants $0, d_1, \sum_{k=1}^{2} d_k, \sum_{k=1}^{4} d_k$ *et* $\sum_{k=1}^{5} d_k$*. En revanche, aucun événement n'a lieu à l'instant* $\sum_{k=1}^{6} d_k$*, ce dernier terme n'apparaît que pour tenir compte du temps écoulé depuis le dernier événement.*

Si une exécution observée est possible avec ou sans action de haut niveau, il n'est alors pas possible pour un attaquant d'en déduire si une action de haut niveau a été utilisée. Par conséquent, garantir qu'aucune exécution observée n'est possible qu'à la condition d'utiliser une action de haut niveau suffit à garantir que l'attaquant sera incapable de déduire si une action de haut niveau est utilisée.

Définition 12 (**Non-interférence avec Observation partielle** (POS-NNI))**.** *Un automate de sécurité* $\mathcal{A}$ *vérifie la POS-NNI par rapport à la fonction d'observation* $\mathcal{O}$ *si et seulement si* $\mathcal{O}(\mathcal{A}) = \mathcal{O}(\mathcal{A}_{\setminus \Sigma_{\mathsf{haut}}})$

On montre maintenant que la POS-NNI est indécidable en la réduisant à l'universalité du langage temporisé, celle-ci étant elle même indécidable [1]. Nous procédons de la manière suivante : D'abord nous définissons une transformation $\mathcal{T}$ sur les automates, préservant le langage et permettant de définir une fonction d'observation telle que l'ensemble des exécutions observées sur $T(\mathcal{A})$ est en bijection avec le langage de $\mathcal{A}$. Nous définissons ensuite pour tout automate $\mathcal{A}$ un automate $\mathcal{A}^*$ comme l'union de $\mathcal{T}(\mathcal{A})$ avec $\mathcal{T}(\mathcal{U})$ ($\mathcal{U}$ étant un automate universel) tel que $\mathcal{A}^*_{\setminus \Sigma_{\mathsf{haut}}} = \mathcal{T}(\mathcal{A})$. $\mathcal{A}^*$ ne vérifie alors la POS-NNI que si et seulement $\mathcal{A}$ a un langage universel.

Définition 13 (Transformation $\mathcal{T}$)**.** *Soit un automate* $\mathcal{A} = (\Sigma, L, l_0, \mathbb{X}, I, E)$*, on définit* $\mathcal{T}(\mathcal{A}) = (\Sigma, L_{\mathcal{T}(\mathcal{A})}, l_{0_{\mathcal{T}(\mathcal{A})}}, \mathbb{X}, I_{\mathcal{T}(\mathcal{A})}, E_{\mathcal{T}(\mathcal{A})})$ *par :*

- $l_{0_{\mathcal{T}}} = (l_0, \epsilon, 0)$
- $L_{\mathcal{T}} = \{L\} \times \{\Sigma \cup \{\epsilon\}\} \times \{0;1\}$
- $\forall l \in L, \forall a \in \Sigma, \forall i \in \{0,1\}, (l,a,i) \in L_{\mathcal{T}(\mathcal{A})} \wedge I_{\mathcal{T}(\mathcal{A})}((l,a,i)) = I(l)$
- $E_{\mathcal{T}(\mathcal{A})}$ *le plus petit ensemble tel que* $\forall e = (l,g,a,R,l') \in E$:
  - $\forall b \in (\Sigma \cup \{\epsilon\}), ((l,b,0),g,a,R,(l',a,1)) \in E_{\mathcal{T}(\mathcal{A})}$
  - $\forall b \in (\Sigma \cup \{\epsilon\}), ((l,b,1),g,a,R,(l',a,0)) \in E_{\mathcal{T}(\mathcal{A})}$

L'idée de la transformation $\mathcal{T}$ est de préserver le langage tout en augmentant le nombre de localités en séparant celles-ci en triplets (l,σ,b). Quand une transition associée à l'action σ est utilisée, l'automate $\mathcal{T}(\mathcal{A})$ atteint nécessairement une localité de la forme $(\bullet,\sigma,\bullet)$. Le paramètre b est une variable binaire permettant d'assurer l'absence de boucle dans $\mathcal{T}(\mathcal{A})$, toute transition disponible depuis une localité de la forme $(\bullet,\bullet,0)$ aura pour cible une localité de la forme $(\bullet,\bullet,1)$ et vice-versa. De cette façon, nous pouvons définir une fonction d'observation sur $\mathcal{T}(\mathcal{A})$ telle que chaque événement observé peut être associé à l'action utilisée.

Proposition 1. *Soit un automate* $\mathcal{A}$ *défini sur l'alphabet* Σ *et sa transformée* $\mathcal{T}(\mathcal{A})$*, les propriété suivantes sont vérifiées :*

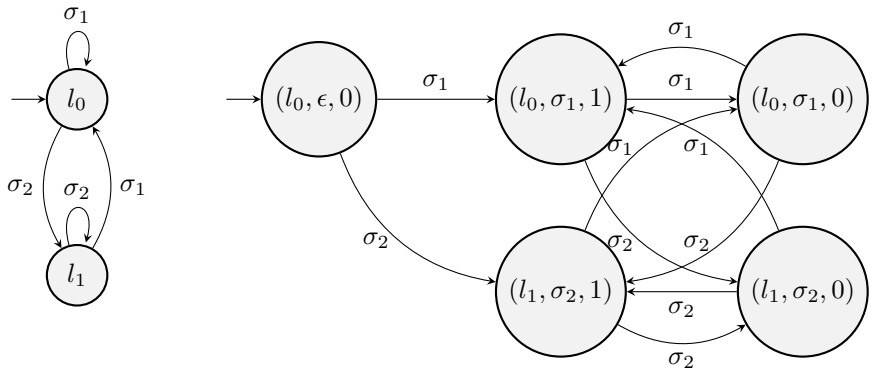

Figure 4: Un automate temporisé et sa transformée par $\mathcal{T}$ (les contraintes temporelles sont omises)

i) $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{T}(\mathcal{A}))$

ii) $\forall((l_s, \sigma_s, j_s), g, a, R, (l_t, \sigma_t, j_t)) \in E_{\mathcal{T}(\mathcal{A})}, a = \sigma_t$

iii) $\forall((l_s, \sigma_s, j_s), g, a, R, (l_t, \sigma_t, j_t)), \in E_{\mathcal{T}(\mathcal{A})}, j_s \neq j_t$

iv) Il n'y a aucune boucle dans $E_{\mathcal{T}(\mathcal{A})}$

Théorème 2. *La POS-NNI est indécidable.*

Esquisse de preuve. Soit un automate $\mathcal{A}$ défini sur l'alphabet Σ, on définit un automate trivialement universel $\mathcal{U} = (\Sigma, l_u, l_u, \emptyset, I(l_u) = vrai, (l_u, \emptyset, \Sigma, \emptyset, l_u))$. On définit alors l'automate de sécurité $\mathcal{A}^*$ qui est l'union de $\mathcal{T}(\mathcal{A})$ avec $\mathcal{T}(\mathcal{U})$ telle qui l'unique action de haut niveau est $(l_{0_{\mathcal{T}(\mathcal{A})}}, , \Sigma_{\mathsf{haut}}, , l_{0_{\mathcal{T}(\mathcal{U})}})$ et dont la localité d'origine est $l_{0_{\mathcal{T}(\mathcal{A})}}$. A cette automate on associe une fonction d'observation telle que toutes les localités sont distinguables des autres à l'exceptions de $l_{0_{\mathcal{T}(\mathcal{A})}}$ et $l_{0_{\mathcal{T}(\mathcal{U})}}$ qui partagent la même observation. De cette façon, l'action de haut niveau n'apparaît pas dans les exécutions observées (elle ne correspond à aucun événement). Clairement, $\mathcal{A}^*$ a le langage de $\mathcal{U}$ lorsqu'il n'est pas restreint aux actions de bas niveau et le langage de $\mathcal{A}$ sinon. De par l'absence de boucle et la correspondance entre l'observation courante et la dernière action de Σ utilisée (Proposition 1), chaque transition de "bas niveau" correspond à un événements observable. Puisque les événement peuvent tous être associés aux actions et que les dates absolues des événements sont mémorisés par l'attaquant, on a donc une bijection entre langage temporisé de $\mathcal{A}^*$ et son ensemble d'exécutions observées. Cette bijection implique que $\mathcal{A}^*$ vérifie la POS-NNI si et seulement $\mathcal{A}$ est universel.

□

5 Modèle d'attaque à mémoire bornée

Nous considérons maintenant un modèle d'attaque ayant une mémoire bornée, ce qui parait être une hypothèse raisonnable.

Un tel modèle d'attaque pourrait choisir entre plusieurs politiques quant au stockage des informations au cours d'une exécution. En première approche, nous nous intéressons à un modèle d'attaque utilisant une politique "premier entré - premier sorti", ne conservant que les

informations des événements les plus récents et effaçant la plus ancienne donnée chaque fois qu'une nouvelle est disponible. Formellement, nous modélisons cette politique en considérant que l'attaquant observe le suffixe de l'exécution observée (pour lequel le temps est relatif à l'information sauvegardée la plus ancienne).

Définition 14 (**Exécutions observées de mémoire k**). *Soit une fonction d'observation $\mathcal{O}$, une exécution ρ et son exécution observée associée $\mathcal{O}(\rho) = (o_1, t_1 = 0)(o_2, t_2)\dots(o_n, t_n)(o_n, t_{n+1})$, l'exécution observée de mémoire k associée à ρ, notée $\mathcal{O}_k(\rho)$, est la séquence définie par :*

si $n \leq k$**:** $\mathcal{O}_k(\rho) = \mathcal{O}(\rho)$

si $n > k$**:** $\mathcal{O}_k(\rho) = (o_{n-k}, 0)(o_{n-k+1}, t_{n-k+1} - t_{n-k})\dots(o_n, t_n - t_{n-k})(o_n, t_{n+1} - t_{n-k})$

On note $\mathcal{O}_k(\mathcal{A})$ l'ensemble des exécutions observées de mémoire k pour toute exécution de $\mathcal{A}$

Comme c'était le cas pour les exécutions observées (avec mémoire infinie), les exécutions observées de mémoire k sont des séquences d'au plus $k+1$ événements. Ici aussi, cela est dû au fait que le modèle d'attaque tient compte du temps écoulé depuis le dernier événement.

Exemple 2 (Une exécution observée et son exécution observée de mémoire 3 associée). *En reprenant l'exemple 1, soit une exécution ρ associée à une exécution observée : $\mathcal{O}(\rho) = (o_1, 0)(o_2, d_1{+}d_2)(o_1, d_1{+}d_2{+}d_3)(o_3, \sum_{k=1}^{5} d_k)(o_3, \sum_{k=1}^{6} d_k)$ On a $\mathcal{O}_3(\rho) = (o_2, 0)(o_1, d_3)(o_3, \sum_{k=3}^{5} d_k)(o_3, \sum_{k=3}^{6} d_k)$.*

On définit alors une nouvelle propriété de non-interférence adaptée à ce modèle d'attaque ayant une mémoire finie :

Définition 15 (**Non-interférence avec observation partielle avec mémoire k** (k-POS-NNI)). *Un automate de sécurité $\mathcal{A}$ vérifie la k-POS-NNI par rapport à la fonction d'observation $\mathcal{O}$ si et seulement si $\mathcal{O}_k(\mathcal{A}) = \mathcal{O}_k(\mathcal{A}_{\setminus \Sigma_{\text{haut}}})$*

5.1 Exemple d'approche de la 2-POS-NNI

Une manière de décider la 2-POS-NNI est d'ajouter 2 horloges (x_1 et x_2) servant à modéliser les durées écoulées depuis chacun des 2 derniers événements observées par le modèle d'attaque. On synchronise alors sur les actions associées aux événements observables de l'automate de sécurité pour lequel on veut décider la 2-POS-NNI avec un automate représentant le contenu de la mémoire de l'attaquant. Lorsque les 2 derniers événements observables se produisent, les horloges x_1 et x_2 sont réinitialisées l'une après l'autre de manière à garder la trace des durées écoulées depuis ces deux événements.

Un automate ayant 2 observations A et B, muni d'une horloge x (à gauche) et l'automate correspondant à l'attaquant muni de 2 horloges x_1 et x_2 (à droite) sont donnés sur la figure. 5. Les localités l_A et $l_{A'}$ sont observées comme A alors que la localité l_B est observée comme B.

Les automates présentés sur la figure 5 illustrent la façon de réduire la 2-POS-NNI à un problème d'atteignabilité d'état. Par exemple, si l'état $(\ell_{(A,B)}; x_1 = 2.5, x_2 = 0.5)$ est atteignable par l'automate d'attaque après synchronisation, cela implique qu'il existe une exécution au cours de laquelle 2 unités de temps s'écoulent dans l'observation A suivi d'une demi unité de temps écoulée dans l'observation B. Réciproquement, pour toute exécution comprenant n événements observés (avec $n > 2$), il existe une exécution sur l'automate d'attaque cyclant sur la localité `Prefix` $n-2$ fois avant d'atteindre les localités $\ell^*_{(A)}$ suivie de $\ell_{(A,B)}$ (ou $\ell^*_{(B)}$ suivie de $\ell_{(B,A)}$), réinitialisant les horloges x_1 et x_2 au moment d'atteindre ces localités. La localité $\ell_{(A)}$ sert à tenir compte des exécutions comprenant moins de 2 événements observées. Elle permet

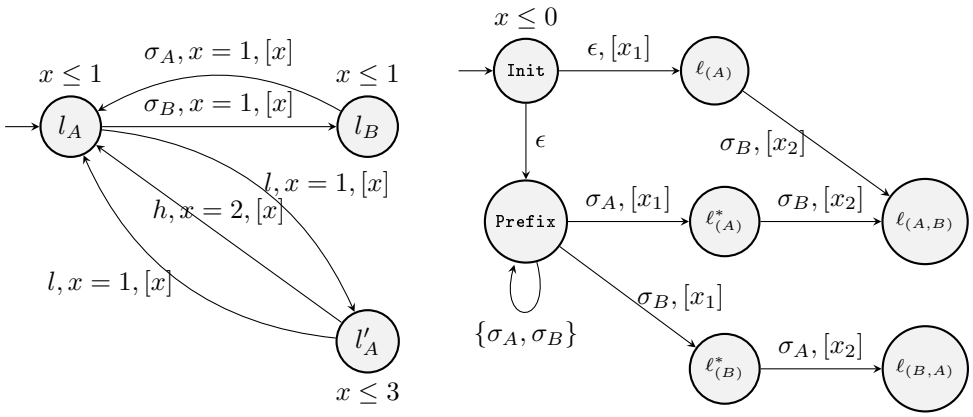

Figure 5: Un automate de sécurité et l'automate d'attaque de mémoire 2 ([] représente les réinitialisation d'horloges)

de "passer" l'étape de cycle sur `Prefix`Cependant, pour une exécution de comprenant $n \geq 2$ événements observés, l'exécution correspondante sur l'automate d'attaque pourrait cycler n ou $n-1$ fois sur la localité `Prefix` et se terminer sur les localités `Prefix`, $\ell^*_{(A)}$ ou $\ell^*_{(B)}$ au lieu de $\ell_{(A,B)}$ ou $\ell_{(B,A)}$. C'est pourquoi les valuations d'horloges possibles sur ces localités (`Prefix` ainsi que les localités marqués d'une astérisque) ne correspondent pas nécessairement à des exécutions observées de mémoire 2 possibles dans l'automate dont on veut garantir la sécurité, ces valuations ne sont donc pas prises en compte.

5.2 Décider la k-POS-NNI

Dans cette section, nous étendons et formalisons la construction proposées dans la section précédente à toute taille de mémoire k et à toute fonction d'observation $\mathcal{O}$.

Soit un automate de sécurité $\mathcal{A} = (\Sigma_{\mathsf{bas}} \cup \Sigma_{\mathsf{haut}}, L, l_0, \mathbb{X}, I, E)$ associé à une fonction d'observation $\mathcal{O} : L \mapsto \Omega$ et à une mémoire d'attaquant k. On définit Ω^j l'ensemble des séquences d'observations (éléments de Ω) de longueurs j.

On définit alors l'automate d'attaque $\mathcal{B}_k = (\{\epsilon\} \cup \Sigma_\Omega, L_{\mathcal{B}_k}, \texttt{Init}, Y, I_{\mathcal{B}_k}, E_{\mathcal{B}_k})$ avec :

- $\Sigma_\Omega = \{\sigma_o \,|\, o \in \Omega\}$
- $L_{\mathcal{B}_k} = \{\texttt{Init}\} \cup \{\texttt{Prefix}\} \cup \{\ell^*_\omega \,|\, \omega \in \Omega^j, j \in [\![1, k-1]\!]\} \cup \{\ell_\omega \,|\, \omega \in \Omega^j, j \in [\![1, k]\!]\}$
- $I_{\mathcal{B}_k}(\texttt{Init}) = (Y = \mathbf{0}), I_{\mathcal{B}_k}(l \neq \texttt{Init}) = \texttt{vrai}$
- $Y = (y_i)_{i \in [\![1,k]\!]}$, où y_i sont de nouvelles horloges, absentes de $\mathcal{A}$.
- $E_{\mathcal{B}_k}$ contient les transitions suivantes :

  (a) initialisation $(\texttt{Init}, \emptyset, \epsilon, [y_1], \ell_{\mathcal{O}(l_0)})$

  (a) événement $\forall j \in [\![2, k]\!], \forall \omega \in \Omega^j, (\ell_{\omega[1,j-1]}, \emptyset, \sigma_{\omega[j]}, [y_j], \ell_\omega) \in E_{\mathcal{B}_k}$

  (b) initialisation $(\texttt{Init}, \emptyset, \epsilon, \emptyset, \texttt{Prefix})$

  (b) événement préfixe $(\texttt{Prefix}, \emptyset, \Sigma_\Omega, \emptyset, \texttt{Prefix})$

  (b) premier événement $\forall o \in \Omega, (\texttt{Prefix}, \emptyset, \sigma_o, [y_1], \ell^*_o) \in E_{\mathcal{B}_k}$

(b) événement intermédiaire $\forall j \in [\![2, k-1]\!], \forall \omega \in \Omega^j, (\ell^*_{\omega[1,j-1]}, \emptyset, \sigma_{\omega[j]}, [y_j], \ell^*_\omega) \in E_{\mathcal{B}_k}$

(b) dernier événement $\forall \omega \in \Omega^k, (\ell^*_{\omega[1,k-1]}, \emptyset, \sigma_{\omega[k]}, [y_k], \ell_\omega) \in E_{\mathcal{B}_k}$

$\mathcal{B}_k$ est simplement la généralisation de l'automate d'attaque proposé sur l'exemple 5. Sur l'exemple, la transition **(a) initialisation** est celle allant de la localité `Init` à la localité $\ell_{(A)}$. Elle sert pour les exécutions de k événements ou moins (En comptant l'initialisation du système comme un événement); La transition **(b) initialisation** est celle allant de la localité `Init` à la localité `Prefix`. Elle est utilisée dans le cas des exécutions de plus de k événements; Les transitions allant de `Prefix` à $\ell^*_{(A)}$ ou $\ell^*_{(B)}$ sont des transitions **(b) premier événement**; La transition allant de $\ell^*_{(A)}$ (respectivement : $\ell^*_{(B)}$) à $\ell_{(A,B)}$ (respectivement: $\ell_{(B,A)}$) est une transition **(b) dernier événement**; Les transitions de type **(b) événement intermédiaire** sont absentes de l'exemple et n'apparaissent que dans les cas $k \geq 3$ où des localités seraient par exemple ajoutées entre $\ell^*_{(A)}$ (respectivement: $\ell^*_{(B)}$) et $\ell_{(A,B)}$ (respectivement : $\ell_{(B,A)}$) afin de tenir compte de la plus grande mémoire de l'attaquant. Enfin, les transitions de type **(a) événement** correspondent à la transition allant de $\ell_{(A)}$ à $\ell_{(A,B)}$. De manière générale, les transitions **(a)** servent à tenir compte des exécutions courtes (ayant moins d'événements que ce que l'attaquant serait capable de mémoriser) tandis que les transitions **(b)** servent dans le cas des exécutions plus longues (ayant plus d'événements que l'attaquant est capable d'en mémoriser), ces dernières modélisent "l'oubli" d'informations par un bouclage sur la localité `Prefix` de manière à associer les derniers k événements aux horloges $(y_i)_{i \in [\![1,k]\!]}$.

$\mathcal{A}$ et $\mathcal{B}_k$ n'utilisant pas les mêmes actions (les actions pertinentes pour l'attaquant étant celles associées à des événements observables), on définit le ϵ-automate $\mathcal{A}^{\mathcal{O}} = (\{\epsilon\} \cup \Sigma_\Omega, L, l_0, \mathbb{X}, E^{\mathcal{O}})$ tel que $\mathcal{O}_k(\mathcal{A}) = \mathcal{O}_k(\mathcal{A}^{\mathcal{O}})$ comme suit :

$$\forall e = (l, g, a, R, l') \in E :$$
$$\mathcal{O}(l) = \mathcal{O}(l') \implies (l, g, \epsilon, R, l') \in E^{\mathcal{O}}$$
$$\mathcal{O}(l) \neq \mathcal{O}(l') \implies (l, g, \sigma_{\mathcal{O}(l')}, R, l') \in E^{\mathcal{O}}$$

On note $\mathcal{A}||_\Sigma\mathcal{B}$ le produit synchronisé à la Arnold Nivat ([5]) de $\mathcal{A}$ avec $\mathcal{B}$ sur l'ensemble d'action Σ.

On peut maintenant formaliser l'équivalence entre les états atteignables de $\mathcal{A}||_\Sigma\mathcal{B}$ et les exécutions observées de $\mathcal{O}_k(\mathcal{A})$:

Lemme 2. *$(o_1, t_1 = 0)(o_2, t_2)\dots(o_j, t_j)(o_j, t_{j+1}) \in \mathcal{O}_k(\mathcal{A})$ si et seulement si $\exists l \in L, \exists \nu$ une valuation sur $Y \cup \mathbb{X}$ telle que $\forall i \in [\![1, j]\!], \nu(y_i) = t_{j+1} - t_i$ et $((l, \ell_{(o_1,\dots,o_j)}), \nu) \in Q^{\mathcal{A}^{\mathcal{O}}||_{\Sigma_\Omega}\mathcal{B}_k}$.*

Tout comme pour la St-NNI, la k-POS-NNI n'est pas résolue directement par des équivalences sur les graphes des régions lorsqu'une borne sur les horloges n'est pas clairement définie. La décidabilité de la k-POS-NNI repose donc ici aussi sur le résultat de *Comon et Jurski*.

Théorème 3. *$\forall k \in \mathbb{N}$, la k-POS-NNI associée à une fonction d'observation $\mathcal{O}$ est décidable.*

Preuve. Soit un automate de sécurité $\mathcal{A} = (\Sigma_{\mathsf{haut}} \cup \Sigma_{\mathsf{bas}}, L, l_0, X, I, E)$, une fonction d'observation $\mathcal{O}$ et une mémoire d'attaquant k, on peut produire une formule $\phi_{(l_0,init),(l,\ell_\omega)}(\mathbf{0}, (y_1, \dots, y_k, x_1, \dots, x_{|X|}))$ (respectivement : $\phi^{\setminus\Sigma_{\mathsf{haut}}}_{(l_0,init),(l,\ell_\omega)}(\mathbf{0}, (y_1, \dots, y_k, x_1, \dots, x_{|X|}))$) qui se "vraie" si et seulement si $((l, \ell_\omega), \nu) \in Q^{\mathcal{A}^{\mathcal{O}}||_{\Sigma_\Omega}\mathcal{B}_k}$ (respectivement : $((l, \ell_\omega), \nu) \in Q^{\mathcal{A}^{\mathcal{O}}_{\setminus\Sigma_{\mathsf{haut}}}||_{\Sigma_\Omega}\mathcal{B}_k}$) tels que $\forall y'_i \in Y, \nu(y'_i) = y_i \wedge \forall x'_i \in X, \nu(x'_i) = x_i$ On définit alors la formule $F = \forall\omega\forall y((\exists l \exists x \phi_{(l_0,init),(l,\ell_\omega)}(\mathbf{0}, (y, x))) \implies$

$(\exists l' \exists x' \phi^{\setminus \Sigma_{\text{haut}}}_{(l_0,init),(l,\ell_\omega)}(\mathbf{0},(y,x))))$. F est équivalente à la propriété $\forall \omega \in \bigcup_{j=1}^{k} \Omega^j \forall y \in \mathbb{R}^k (\exists l \in L, \exists x \in \mathbb{R}^{|X|} telsque(l,\ell_\omega,\nu) \in Q^{\mathcal{A}^{\mathcal{O}} || \Sigma_\Omega \mathcal{B}_k} s.t \nu(Y) = y \wedge \nu(X) = x) \implies (\exists l' \in L, \exists x' \in \mathbb{R}^{|X|} telsque(l',\ell_\omega,\nu') \in Q^{\mathcal{A}^{\mathcal{O}}_{\setminus \Sigma_{\text{haut}}} || \Sigma_\Omega \mathcal{B}_k} s.t \nu'(Y) = y \wedge \nu'(X) = x')$ D'après le Lemme 2, F est donc équivalente à : $\forall o \in \mathcal{O}_k(\mathcal{A}), o \in \mathcal{O}_k(\mathcal{A}^{\setminus \Sigma_{\text{haut}}})$. Pour tout automate de sécurité $\mathcal{A}$ on a $\Psi(\mathcal{A}^{\setminus \Sigma_{\text{haut}}}) \subseteq \Psi(\mathcal{A})$, ce qui implique $\mathcal{O}_k(\mathcal{A}^{\setminus \Sigma_{\text{haut}}}) \subseteq \mathcal{O}_k(\mathcal{A})$ On a donc montré que pour tout automate de sécurité associé à une fonction d'observation et une mémoire d'attaquant, on peut produire une formule décidable qui sera vraie si et seulement si $\mathcal{O}_k(\mathcal{A}) = \mathcal{O}_k(\mathcal{A}^{\setminus \Sigma_{\text{haut}}})$, c'est à dire si et seulement si l'automate vérifie la k-POS-NNI. □

5.3 Automates à observations de durées bornées (FDO)

Bien que la k-POS-NNI soit décidable, la formule utilisée pour décider la propriété sur un automate donné peut s'avérer coûteuse à produire ainsi qu'à vérifier. On introduit donc une classe d'automates pour laquelle la k-POS-NNI est résolue plus facilement, il s'agit des automates associées à des fonctions d'observations telle que la durée entre deux événements consécutifs d'une exécution est bornée.

Définition 16. *Un automate $\mathcal{A}$ est à observations de durées bornées (FDO) par rapport à une fonction d'observation $\mathcal{O}$*[1] *si et seulement si $\exists M \in \mathbb{R}_{\geq 0}$ tel que $\forall \rho \in \Psi(\mathcal{A})$ avec $\mathcal{O}(\rho) = (o_0, t_0 = 0)(o_1,t_1)\cdots(o_n,t_n), \forall 0 \leq k < n$, on a $t_{k+1} - t_k \leq M$.*

On appelle M la borne d'observation *de $\mathcal{A}$.*

Dans le cas des automates pour lesquels le temps ne peut pas s'écouler indéfiniment sans qu'aucun événement observable ne se produise, les horloges ajoutée pour modéliser la mémoire de l'attaquant (celles ajoutées par le produit avec $\mathcal{B}_k$) deviennent bornées. Cela permet d'assurer l'équivalence entre régions atteignables du graphe des régions et états atteignables de l'automate en choisissant une constante dans la définition des régions telle que les horloges de l'attaquant n'excèdent jamais cette borne.

Proposition 2. *Soit un automate $\mathcal{A}$ associé à une fonction d'observation $\mathcal{O}$. L'appartenance de $\mathcal{A}$ à l'ensemble des automates FDO pour $\mathcal{O}$ est décidable.*

Corollaire 1. *Soit une fonction d'observation $\mathcal{O}$ associée à l'automate $\mathcal{A}$ muni de x horloges, c étant la plus grande constante utilisée dans ses contraintes temporelles et L le cardinal de son ensemble de localités. Si $\mathcal{A}$ est FDO par rapport à $\mathcal{O}$, alors sa borne d'observation M vérifie la relation suivante :*

$$M \leq x!(2c+2)^x L$$

Corollaire 2. *Soit une fonction d'observation $\mathcal{O}$ associée à un automate $\mathcal{A}$. Si $\mathcal{A}$ est FDO par rapport à $\mathcal{O}$, alors sa borne d'observation M peut être calculée.*

5.4 Décider la k-POS-NNI pour les automates FDO

Dans cette section, on montre que décider la k-POS-NNI pour les automates FDO est un problème PSPACE-complet.

Théorème 4. *$\forall k \in \mathbb{N}$, la k-POS-NNI associée à la fonction d'observation $\mathcal{O}$ pour les automates FDO par rapport à $\mathcal{O}$ peut être décidée par une construction sur les graphes des régions.*

[1] On dit que l'automate est FDO par rapport à $\mathcal{O}$.

Bien que le calcule des graphes des régions de $\mathcal{A}||_\Sigma\mathcal{B}$ et $\mathcal{A}_{\setminus\Sigma_{\text{haut}}}||_\Sigma\mathcal{B}$ demanderait une mémoire importante si la décision de la k-POS-NNI était implémentée telle quelle, un algorithme PSPACE existe.

Théorème 5. *Avec k représenté en unaire, la k-POS-NNI pour les automates FDO est PSPACE-complète.*

Esquisse de preuve. On montre d'abord la PSPACE-appartenance, pour cela on utilise la PSPACE-appartenance de l'atteignabilité d'une région dans le graphe des régions d'un automate. Puisque $\mathcal{B}_k$ est défini à partir de $\mathcal{O}$ et k, il n'est pas nécessaire de le calculer explicitement pour explorer le produit avec $\mathcal{A}$, il suffit en réalité de comparer les observations des localités source et cible de chaque transition afin d'ajouter où non à la mémoire de l'attaquant l'observation de la localité cible. La séquence d'observation devant être mémorisée par l'attaquant, un terme linéaire par rapport à k apparaît dans la représentation de l'état sur la mémoire, d'où la nécessité pour k d'être représenté en unaire afin de préserver une procédure PSPACE sur la taille de la donnée d'entrée. Les valeurs des horloges y étant bornées (1), celles-ci n'entraînent qu'une augmentation polynomiale de l'espace nécessaire pour mémoriser les état. On peut alors explorer de manière non-déterministe l'atteignabilité de chaque région sans avoir à calculer explicitement $\mathcal{B}_k$ ni le graphe des régions. Pour montrer que le problème est PSPACE-difficile on le réduit à l'atteignabilité des localités dans les automates temporisés, qui est elle même PSPACE-difficile. Soit un automate et une localité "cible", on définit son automate "augmenté" d'une fonction d'observation auquel on ajoute un localité distinguable de toutes les autres. De plus, on assure que cette localité soit atteignable depuis la localité initiale par une transition de haut niveau mais aussi par une transition de bas niveau depuis la localité "cible". D'autres modifications doivent être apportées pour garantir que l'automate "augmenté" soit FDO. L'observation unique ne pouvant être présente sur des exécutions observées de bas niveau que si et seulement si la localité cible est atteignable, cela conclut la preuve. □

6 Discussion et travaux futurs

Nous avons introduit une nouvelle propriété de non-interférence basée sur l'état. Nous pensons que cette propriété est plus réaliste que celles précédemment étudiées dans la littérature car elle modélise un attaquant plus "malin" et muni d'un pouvoir d'observation plus raisonnable. Nous avons montré que cette propriété est indécidable dans le cas général. Nous avons cependant montré que celle-ci devient décidable lorsque la mémoire de l'attaquant est bornée. De plus nous avons fourni une classe d'automates pour lesquels la propriété est PSPACE-complète dans le cas d'une mémoire finie, nous avons également montré comment décider l'appartenance à cette classe. Nous avons également démontré la décidabilité de la St-NNI dans le cas général. Comme nous l'avons montré, le problème n'était en fait résolu que dans le cas d'horloges bornées.

De futurs travaux consisteraient à étudier d'autres politiques de stockage de l'information pour l'attaquant. Nous souhaitons également produire des algorithmes plus efficaces pour décider la k-POS-NNI dans le cas des automates FDO, par exemple en se basant sur les zones représentées par des DBM plutôt que les régions. Enfin nous souhaitons également compléter l'étude de la k-POS-NNI en determinant la complexité de sa vérification dans le cas général.

7 Remerciements

Les auteurs remercient Patricia Bouyer pour les discussions fructueuses sur le sujet, notamment sur l'utilisation des travaux de Comon et Jurski dans ce contexte.

References

[1] Rajeev Alur and David L Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[2] Ikhlass Ammar, Yamen El Touati, Moez Yeddes, and John Mullins. Bounded opacity for timed systems. *Journal of Information Security and Applications*, 61:102926, 2021.

[3] Étienne André and Aleksander Kryukov. Parametric non-interference in timed automata. In *2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS)*, pages 37–42. IEEE, 2020.

[4] Étienne André, Didier Lime, Dylan Marinho, and Jun Sun. Guaranteeing timed opacity using parametric timed model checking. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(4):1–36, 2022.

[5] André Arnold. Finite transition systems. *Prentice Hall*, 1994.

[6] Roberto Barbuti and Luca Tesei. A decidable notion of timed non-interference. *Fundamenta Informaticae*, 54(2-3):137–150, 2003.

[7] Gilles Benattar, Franck Cassez, Didier Lime, and Olivier H Roux. Control and synthesis of non-interferent timed systems. *International Journal of Control*, 88(2):217–236, 2015.

[8] Andrew Bortz and Dan Boneh. Exposing private information by timing web applications. In *Proceedings of the 16th international conference on World Wide Web*, pages 621–628, 2007.

[9] Jeremy Bryans, Maciej Koutny, Laurent Mazare, and Peter Y. A. Ryan. Opacity generalised to transition systems. *International Journal of Information Security*, 7(6):421–435, 2008.

[10] Hubert Comon and Yan Jurski. Timed automata and the theory of real numbers. In *CONCUR*, volume 99, pages 242–257. Springer, 1999.

[11] Edward W. Felten and Michael A. Schneider. Timing attacks on web privacy. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, CCS '00, page 25–32, New York, NY, USA, 2000. Association for Computing Machinery.

[12] Guillaume Gardey, John Mullins, and Olivier H Roux. Non-interference control synthesis for security timed automata. *Electronic Notes in Theoretical Computer Science*, 180(1):35–53, 2007.

[13] Christopher Gerking, David Schubert, and Eric Bodden. Model checking the information flow security of real-time systems. In *Engineering Secure Software and Systems: 10th International Symposium, ESSoS 2018, Paris, France, June 26-27, 2018, Proceedings 10*, pages 27–43. Springer, 2018.

[14] Joseph A Goguen and José Meseguer. Unwinding and inference control. In *1984 IEEE Symposium on Security and Privacy*, pages 75–75. IEEE, 1984.

[15] Paul C Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Annual International Cryptology Conference*, pages 104–113. Springer, 1996.

[16] Robert Kotcher, Yutong Pei, Pranjal Jumde, and Collin Jackson. Cross-origin pixel stealing: timing attacks using css filters. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1055–1062, 2013.

[17] Anthony Spriet, Didier Lime, and Olivier H. Roux. Timed non-interference under partial observability and bounded memory. In *21st International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2023)*, Lecture Notes in Computer Science. Springer, September 2023.

[18] Lingtai Wang and Naijun Zhan. Decidability of the initial-state opacity of real-time automata. In *Symposium on Real-Time and Hybrid Systems: Essays Dedicated to Professor Chaochen Zhou on the Occasion of His 80th Birthday*, pages 44–60. Springer, 2018.

[19] Volker Weispfenning. Mixed real-integer linear quantifier elimination. In *Proceedings of the 1999 international symposium on Symbolic and algebraic computation*, pages 129–136, 1999.