Utilisation de la corrélation pondérée dans un processus de détection d'intrusions

Fabien Autrel[1], Salem Benferhat[2], Frédéric Cuppens[3]

[1]ONERA-CERT, 2 Av. E. Belin, 31055, Toulouse Cedex, France,

[2]CRIL CNRS Université d'Artois Rue Jean Souvraz SP 18 F 62307 Lens Cedex, France

[3] École Nationale Supérieure des Télécommunications de Bretagne, BP 78 - 2, rue de la Châtaigneraie - F-35512 Cesson Sévigné Cedex

email : autrel@cert.fr, benferhat@cril.univ-artois.fr, frederic.cuppens@enst-bretagne.fr

Résumé

En général un attaquant doit réaliser plusieurs actions, organisées en un scénario d'intrusions, afin d'arriver à atteindre ses objectifs. Nous représentons ces actions par leurs pré et post conditions qui correspondent à un ensemble de prédicats logiques ou de négation de prédicats. La pré condition d'une action représente l'état dans lequel doit être le système afin de pouvoir exécuter l'action. La post condition correspond aux effets de l'exécution de l'action sur l'état du système.

Quand un attaquant commence son intrusion, nous pouvons déduire, de l'observation des alertes générées par les SDI (Systèmes de Détection d'Intrusions), plusieurs scénarios d'intrusions possibles en corrélant les actions. Cependant nous ne pouvons pas déterminer quel est le scénario le plus plausible dans l'ensemble des scénarios générés sans une analyse plus poussée. Dans cet article nous proposons de définir une relation d'ordre sur l'ensemble des scénarios générés en pondérant les actions composant un scénario d'attaque.

Mots clefs : Système de Détection d'Intrusion (SDI), corrélation pondérée, scénarios d'intrusion, alertes.

Résumé

Abstract

Generally, an intruder must perform several actions, organized in an intrusion scenario, to achieve his or her malicious objectives. Actions are modeled by their pre and post conditions, which are a set of logical predicates or negations of predicates. Pre conditions of an action correspond to conditions the system's state must satisfy to perform the action. Post conditions correspond to the effects of executing the action on the system's state.

When an intruder begins his intrusion, we can deduce, from the alerts generated by IDSs (Intrusion Detection Systems), several possible scenarios, by correlating attacks, that lead to multiple intrusion objectives. However, with no further analysis, we are not able to decide which are

the most plausible ones among the possible scenarios. We propose in this paper to define an order over the possible scenarios by weighting the correlation relations between successive attacks composing the scenarios. These weights reflect to what level executing some actions are necessary to execute some action B. We will see that to be satisfactory, the comparison operator between two scenarios must satisfy some properties.

Keywords : Intrusion Detection System (IDS), weighted correlation, intrusion scenario, alerts.

I. Introduction

Le principal objectif de la sécurité des systèmes d'informations est de concevoir et développer des systèmes qui soient conformes aux spécifications d'une politique de sécurité. Une politique de sécurité est un ensemble de règles qui spécifient les autorisations, les interdictions et les obligations des agents (utilisateurs et applications) qui peuvent accéder au système. Un intrus (aussi appelé "hacker" ou "cracker") peut être vu comme un agent malveillant essayant de violer la politique de sécurité. Informellement une intrusion est définie comme une tentative délibérée de violer la politique de sécurité.

Parfois l'attaque se résume à une seule action. Par exemple réaliser un déni de service en utilisant l'attaque "ping of death" ne nécessite que l'envoie d'un paquet IP trop long. Cependant des intrusions plus complexes necessitent généralement que plusieurs actions soient effectuées [17, 5, 7, 3]. Prenons l'exemple de l'attaque *Mitnick*. Cette attaque se décompose en deux étapes. La première étape consiste à inonder une machine H, la deuxième étape consiste à établir une connexion avec un serveur S en usurpant l'adresse source afin de faire croire que les messages proviennent de H. Lorsque S envoie le message SYNACK à H, H ne peut pas répondre et l'attaquant n'a plus qu'à envoyer le message ACK afin d'ouvrir illégalement une connexion avec S. Il est à noter que cette attaque constitue probablement la première partie du scénario de l'attaquant dont l'objectif final serait d'obtenir un accès à S en exécutant un *rlogin* par exemple. L'attaque *Mitnick* représente donc ici une première étape d'un scénario plus global. Dans la suite, nous appellerons *scénario d'intrusions* la séquence complète d'actions qui permet à l'intrus d'atteindre ses objectifs.

Dans [12, 13], la notion de corrélation d'attaques, qui permet de reconnaître les différentes étapes d'un scénario d'intrusion, a été définie. Puis une approche, basée sur la corrélation d'attaques, permettant de déterminer si une séquence d'actions corrélées peut aboutir à un objectif d'intrusion (reconnaissance d'intentions malveillantes) a été développée. Cette approche permet de construire un ensemble d'instances de scénarios possibles compatibles avec les observations générées par les SDIs (Systèmes de Détection d'Intrusion).

L'approche proposée dans [12, 13] n'est pas complètement satisfaisante : en effet le nombre de scénarios possibles peut être grand et aucune information n'est donnée à l'administrateur système pour choisir le scénario le plus plausible.

Cet article propose une nouvelle approche de la corrélation d'alertes qui

permet de classer les différents scénarios possibles. Nous distinguons deux types de relations d'influence entre deux actions :

- une relation d'influence positive : l'occurrence de l'action A peut aboutir à la réalisation de l'action B
- une relation d'influence négative : la réalisation de l'action A bloque la réalisation de B

Ensuite nous associons à chaque action A un poids qui représente la plausibilité de réalisation de l'action A sous l'hypothèse que certaines actions (qui peuvent directement influencer la réalisation de A) ont été réalisées. Ces poids nous permettront de comparer et de classer différents scénarios.

La suite de cet article est organisée comme suit : la section 2 présente les techniques de corrélation d'alertes existantes. La section 3 présente une architecture de détection coopérative dans laquelle la corrélation est intégrée. La section 4 introduit la représentation des actions et des scénarios, ces représentations sont des extensions de celles présentées dans [12] prenant en compte la notion de date de détection. La section 5 présente un exemple et illustre la nécessité d'utiliser des méthodes limitant le nombre des scénarios possibles. La section 6 présente la notion de corrélation pondérée et définit le poids de corrélation associé à une action dans un scénario d'attaques. La section 7 présente une méthode qui compare des scénarios d'attaques et qui sélectionne les plus plausibles. La section 8 conclut l'article.

II. Les techniques de corrélation existantes

La notion de corrélation d'alertes dans les systèmes de détection d'intrusion n'est pas nouvelle et plusieurs articles ont déjà proposé une définition de cette notion [17]. Cette section se propose de faire un état de l'art des techniques de corrélation d'alertes.

La définition de corrélation d'alertes donnée dans la littérature fait référence à des problèmes très différents. Par corrélation d'alertes Valdes et Skinner [5] désignent le processus qui consiste à agréger des alertes relatives à l'occurrence d'événements similaires. Leur système appelé EMERALD, construit, à partir des alertes générées par des sondes probabilistes, ce qu'ils appellent des méta alertes. Le principe est d'ajouter une alerte candidate à la méta alerte la plus similaire sous réserve de se situer au-dessus d'un seuil de similarité. Les scénarios composés de plusieurs étapes distinctes sont construits en relaxant la similarité minimale attendue sur la classe associée à une meta alerte. Aucun modèle n'est utilisé pour caractériser une attaque ; outre la liste de ses attributs (IP source, IP cible, ports cibles, etc...), le seul attribut permettant de caractériser une attaque est sa classe. Dans cette optique, ce qu'ils appellent scénario d'alertes est un ensemble de méta alertes similaires au niveau de leurs attributs.

Dans [6], les auteurs proposent un système qui construit des scénarios en temps réel au fur et à mesure que les alertes de détection d'intrusion sont générées. Chaque nouvelle alerte est associée à un groupe existant d'alertes par un système de fusion. Les auteurs appellent ces groupes d'alertes “scénarios”

mais ne donnent pas de définition formelle de la notion de scénario. Les auteurs ajoutent aux informations des alertes données par les SDI la catégorie de l'alerte. Cinq catégories ont été définies : *discovery, scan, escalation, denial-of-service, stealth.* Grâce à cette information supplémentaire, les auteurs peuvent corréler de manière probabiliste deux alertes. Trois quantités sont évaluées afin de déterminer si une alerte doit être ajoutée à un scénario (groupe d'alertes) :

- l_{ij}, qui mesure la probabilité que deux alertes soient corrélées à partir de leur catégorie,
- $\sigma_{ij}(\Delta t)$, qui mesure la distance temporelle entre deux alertes,
- R_{ij} qui mesure la similarité entre les adresses source des alertes.

Ces trois mesures sont définies sur l'intervalle $[0, 1]$ et leur produit donne le résultat final de la mesure. Les auteurs montrent que leur système de fusion regroupe les alertes comme le ferait un analyste à travers des résultats expérimentaux produits sur les données du concours DEFCON (www.defcon.org).

Skinner et coll. [14] proposent de construire des graphes d'attaques pouvant se dérouler sur un réseau donné mais ne s'appuient pas sur les données générées par des SDI. Ils proposent de découvrir les scénarios pouvant être potentiellement utilisés sur un réseau pour compromettre la politique de sécurité. Ils se basent sur une modélisation des attaques par pré et post condition, les pré et post conditions relatives au réseau et à l'attaquant étant séparées, et sur une connaissance des vulnérabilités du réseau pour explorer l'espace des états à l'aide d'un model-checker (NuSMV). Etant donnée une propriété (qui exprime ce que le graphe ne doit pas satisfaire) les auteurs génèrent un graphe d'états. Par exemple les auteurs génèrent le graphe d'attaque permettant à un attaquant d'obtenir sur un réseau donné un accès root sans être détecté. Ce travail expose donc une méthode de découverte de vulnérabilités du réseau, sous forme de scénarios d'attaque étant donnée une configuration. En revanche, il ne s'agit pas d'une technique de détection d'intrusion dans le sens classique du terme, c'est-à-dire explorer les données générées lors de la surveillance d'un système informatique pour détecter des intrusions et/ou les empêcher.

Dans [12] la notion de corrélation désigne le processus permettant de dégager un ensemble d'actions organisées en un scénario d'intrusion dans l'ensemble des alertes de détections d'intrusion. Le langage de modélisation des actions est exposé dans [4]. Chaque action est modélisée par sa pré-condition (conditions requises pour que l'action puisse s'exécuter) et sa post-condition (effets de l'action sur le système). Le moteur de corrélation est aussi capable de palier à la non détection d'une étape d'un scénario grâce à la génération d'hypothèse. Ce mécanisme est aussi utilisé pour pouvoir anticiper les actions de l'attaquant en générant les chemins d'attaques possibles.

La notion de corrélation présentée dans [15] est très proche de la définition de [12]. Les auteurs donnent une modélisation des attaques se basant sur les conditions requises pour exécuter une attaque et sur les possibles conséquences de l'attaque. Ces deux éléments sont représentés par des prédicats. Contrairement à l'approche donnée dans [12], les auteurs n'excluent pas l'utilisation de l'opérateur de disjonction dans l'expression des pré-requis et des conséquences. Ils définissent la notion d'hyper-alerte qui contient le modèle d'une attaque.

Les alertes sont des instances de ces hyper-alertes. Les auteurs précisent qu'une instance d'hyper-alerte peut contenir plusieurs alertes générées par des SDIs, mais ils ne précisent pas par quel mécanisme ces groupes d'alertes sont générés. La condition permettant à deux hyper-alertes d'être corrélées est assez faible, comme dans [12]. En effet deux hyper-alertes sont corrélées si la première contribue à la réalisation de la deuxième de par l'inclusion d'une partie des prédicats de la conséquence de la première alerte dans les prédicats des pré-requis de la deuxième alerte.

Dans [16], Ning et Xu étendent les notions présentées dans [15] afin de dégager des stratégies d'attaques à partir des graphes de corrélation d'hyper-alertes. L'idée est de déduire d'un graphe de corrélation d'hyper-alertes consitué d'instances d'hyper alertes, un graphe de type d'hyper-alertes (correspondant à un graphe d'alertes non instanciées) révélant la stratégie de l'attaque, c'est-à-dire ces différentes étapes, indépendamment du système attaqué. Dans une deuxième partie, les auteurs présentent une méthode pour mesurer la similarité entre deux stratégies d'attaque en mesurant le coût de transformation du graphe d'une stratégie en un autre graphe.

III. La corrélation dans la chaîne de traitement des alertes

La technique de corrélation présentée dans cet article a été conçue dans le cadre d'une certaine architecture (voir figure 1) de détection coopérative ([12]). Afin d'améliorer le taux de détection, plusieurs SDIs sont utilisés dans cette archtecture. L'étape de corrélation des alertes a une position bien définie dans la chaîne de traitement des alertes de détection d'intrusion. En effet, dans le cadre de la détection d'intrusion coopérative, la quantité d'alertes générées peut être très importante et il existe des alertes redondantes. Par alertes redondantes nous désignons le fait que si plusieurs SDIs détectent une attaque, plusieurs alertes relatives au même événement sont émises. Il est donc nécessaire de regrouper les alertes relatives au même événement afin de créer une alerte globale pour ensuite l'envoyer vers le processus de corrélation. Cette étape de regroupement décrite dans [10] est en fait constituée de deux processus, le premier correspondant à l'agrégation d'alertes relatives à l'occurrence d'un même évènement et le second processus étant la fusion des informations contenues dans un groupe d'alertes afin de créer une alerte globale.

Dans cet article nous ne nous intéressons pas à l'étape de la fusion d'alertes (voir figure 1), mais uniquement à l'étape de corrélation d'alertes. La section suivante présente la modélisation de l'intrusion.

IV. Modélisation de l'intrusion

Afin de modéliser le processus d'intrusion, nous étendons les notions définies dans [12]. Nous considérons que l'attaquant a à sa disposition un ensemble

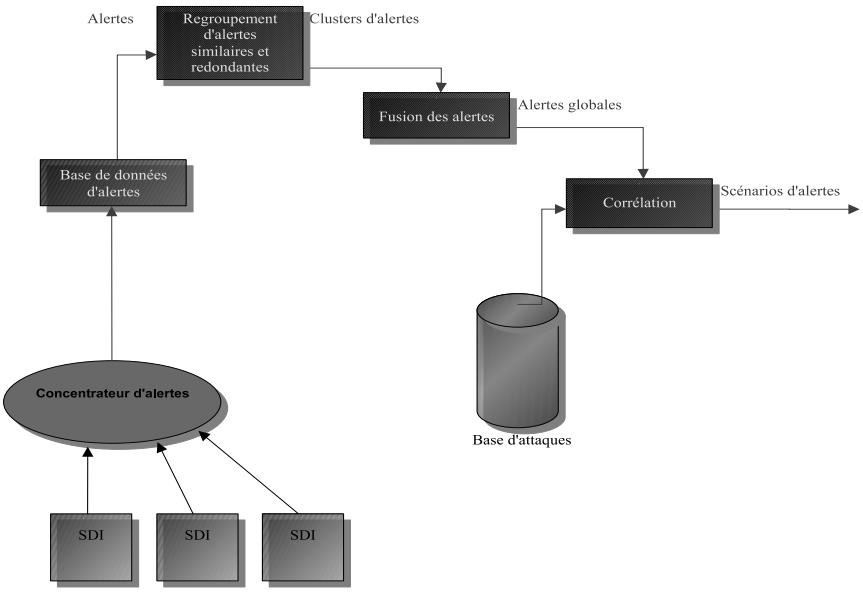

FIG. 1 – Architecture d'un système de détection d'intrusions coopératif

d'actions lui permettant d'atteindre son objectif d'intrusion. Plus précisément il doit trouver un sous ensemble d'actions lui permettant d'atteindre un état du système dans lequel la politique de sécurité est violée, c'est-à-dire un état dans lequel l'objectif d'intrusion est atteint.

L'approche repose donc sur la modélisation d'un ensemble d'actions que les attaquants peuvent exécuter et un ensemble d'objectifs d'intrusion qu'il faut protéger. Les systèmes de détection d'intrusion (SDIs) génèrent des alertes qui correspondent à des instances de ces actions.

Notre but est double :

- Expliquer les actions malveillantes qui ont été observées dans le flux d'alertes (analyse hors ligne).
- Trouver dans le flux d'alertes générées s'il existe des séquences d'instances d'actions permettant d'atteindre un objectif d'intrusion (analyse en ligne).

Les sous-sections suivantes présentent tout d'abord la modélisation des actions, des instances d'action et des objectifs d'intrusion, puis définissent les notions de corrélation d'actions et de scénario d'intrusion.

IV.1. Représentation des actions

Dans [12], les actions sont représentées par leur pré et post conditions, conformément au langage LAMBDA ([4]). La pré condition d'une action représente l'état dans lequel le système doit être afin de pouvoir exécuter l'action. La post

condition exprime les effets de l'action sur l'état du système.

La découverte de liens de corrélation dans un ensemble d'actions fait intervenir la date à laquelle ces actions sont détectées. En pratique nous traitons des alertes datées, associées aux attaques, qui sont totalement ordonnées. L'ordre d'un ensemble d'alertes est imposé par l'intrus qui exécute l'attaque.

Nous proposons deux modélisations. La première modélise les actions pouvant être exécutées par un attaquant indépendamment d'un système informatique donné (actions conceptuelles), et la deuxième modélise les instances d'actions qui seront observées sur un réseau donné par des SDIs (actions observées).

Définition 1 : modélisation des actions Une action A est modélisée par 3 champs :

- $Name(Param_1, Param_2, ..., Param_n)$: expression fonctionnelle représentant le nom de l'action ainsi que ses paramètres.
- Pré-condition : conjonction de prédicats que le système doit satisfaire pour que l'action puisse s'effectuer.
- Post condition : conjonction de prédicats exprimant les effets de l'action sur le système.

Le modèle d'instance d'actions reprend les mêmes champs que précédemment et ajoute un champ *Detectime* afin de dater l'instance. Les paramètres de l'actions sont instanciés.

Définition 2 : modélisation des instances d'actions Une instance d'action A est modélisée par 4 champs :

- $Name(Param_1, Param_2, ..., Param_n)$: expression fonctionnelle représentant le nom de l'action ainsi que ses paramètres instanciés.
- DetectTime : la date à laquelle l'action a été détectée.
- Pré-condition : conjonction de prédicats que le système doit satisfaire pour que l'action puisse s'effectuer.
- Post condition : conjonction de prédicats exprimant les effets de l'action sur le système.

Dans la suite de l'article, $DetectTime(Action_i)$ désigne la date de l'instance d'une action. $Pre(Action_i)$ et $Post(Action_i)$ désignent respectivement les pré et post conditions d'une action.

La figure 2 montre quelques exemples de modélisation d'actions que nous utiliserons plus loin pour définir des scénarios.

IV.2. Représentation des objectifs d'intrusion

Les objectifs d'intrusion sont modélisés par une condition sur l'état du système.

Action $touch(Agent, File)$
Pre : $true$
Post : $file(File)$,
$authorized(Agent, read, File)$,
$authorized(Agent, write, File)$

Action $block(Agent, Printer)$
Pre : $printer(Printer)$,
$physical\_access(Agent, Printer)$,
not$(blocked(Printer))$
Post : $blocked(Printer)$

Action $lpr\ -s(Agent, Printer, File)$
Pre : $printer(Printer)$,
$file(File)$,
$authorized(Agent, read, File)$
Post : $queued(File, Printer)$

Action $remove(Agent, File)$
Pre : $authorized(Agent, write, File)$,
$file(File)$
Post : not $(file(File))$

Action $ln\ -s(Agent, Link, File)$
Pre : not $(file(Link))$,
$file(File)$
Post : $linked(Link, File)$

Action $unblock(Agent, Printer)$
Pre : $printer(Printer)$,
$blocked(Printer)$,
$physical\_access(Agent, Printer)$
Post : not $(blocked(Printer))$

Action $print\text{-}process(Printer, Link)$
Pre : $queued(Link, Printer)$,
$linked(Link, File)$,
not $(blocked(Printer))$
Post : $printed(Printer, File)$,
not $(queued(Link, Printer))$

Action $get\text{-}file(Agent, File, Printer)$
Pre : $printed(Printer, File)$,
$physical\_access(Agent, Printer)$
Post : $read\_access(Agent, File)$

FIG. 2 – Actions utilisées dans le scénario *illegal file access*

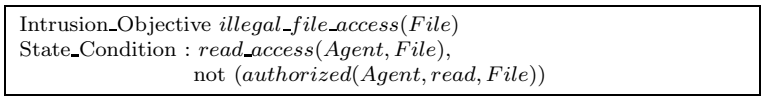

FIG. 3 – Objectif d'intrusion *illegal file access*

Définition 3 : modélisation des objectifs d'intrusion Un objectif d'intrusion O est modélisé par deux champs :

- $Name(Param_1, Param_2, ..., Param_n)$: expression fonctionnelle représentant le nom de l'objectif ainsi que ses paramètres.
- StateCondition : conjonction de prédicats que le système satisfait une fois l'objectif atteint.

Par exemple l'objectif d'intrusion $DOS\_on\_DNS(Host)$ est atteint lorsque les deux prédicats suivants sont satisfaits : $dns\_server(Host)$ et $dos(Host)$. Ces prédicats signifient respectivement que $Host$ est un serveur DNS et qu'il n'est pas disponible. La figure 3 montre un autre exemple de modélisation d'un objectif d'intrusion. L'objectif d'intrusion *illegal file access* est atteint lorsqu'un agent obtient un accès en lecture sur un fichier alors qu'il n'est pas autorisé à le lire.

De la même manière que pour les actions, nous définissons la notion d'instance d'objectif d'intrusion :

Définition 4 : modélisation des instances d'objectifs d'intrusion Un objectif d'intrusion O est modélisé par deux champs :

- $Name(Param_1, Param_2, ..., Param_n)$: expression fonctionnelle représentant le nom de l'objectif ainsi que ses paramètres instanciés.
- StateCondition : conjonction de prédicats que le système satisfait une fois l'objectif atteint.

Notons que la date à laquelle l'objectif d'intrusion est atteint n'est pas incluse. En effet un objectif d'intrusion est atteint une fois qu'un ensemble d'action ont été exécutées et qu'elle ont mis le système dans un certain état. Le moment auquel l'objectif est atteint correspond donc à la date de l'occurence de la dernière action ayant permis d'atteindre l'état du système dans lequel la condition de l'objectif est ateint.

IV.3. Représentation de la connaissance sur le domaine ou l'état du système

La connaissance sur le domaine, ou état du système, est représentée par K et contient les informations disponibles sur le système. Cette connaissance est représentée par un ensemble de prédicats. Par exemple, la connaissance K du domaine peut être égale à $\{file(secret\_file), printer(ppt),$
$physical\_access(bad\_guy, ppt)\}$, qui veut dire que $secret\_file$ est un fichier, ppt est une imprimante et que bad_guy peut accéder à l'imprimante ppt. Ces données pourraient être contenues dans une base de données s'appuyant sur le modèle formel M2D2 défini dans [11].

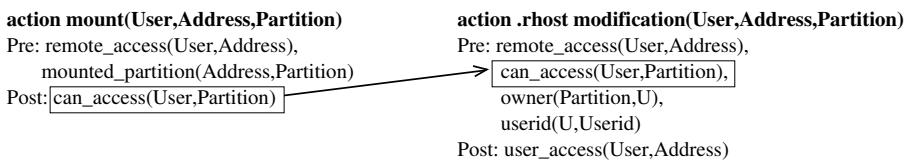

FIG. 4 – Exemple d'influence positive entre deux actions

IV.4. Corrélation des actions

Une fois que l'attaquant a défini son objectif d'intrusion et le scénario qui va lui permettre de l'atteindre, l'ensemble des actions nécessaires pour parvenir à l'objectif est fixé. Du point de vue de la détection d'intrusion, nous devons trouver parmi un nombre élevé d'alertes un ensemble d'alertes permettant d'atteindre un objectif d'intrusion. Afin d'accomplir cette tâche nous définissons le concept de corrélation d'actions. La corrélation d'actions nous permet de dire que si une action A est corrélée avec une action B, alors A peut avoir une influence sur la réalisation de B ([12]).

Soient E et F deux expressions logique [1] ayant la forme suivante :

- $E = expr_{E_1}, expr_{E_2}, ..., expr_{E_m}$
- $F = expr_{F_1}, expr_{F_2}, ..., expr_{F_n}$

où chaque $expr_{E_i}$ (resp. $expr_{F_i}$) est un prédicat ou la négation d'un prédicat, c'est-à-dire que $expr_{E_i}$ (resp. $expr_{F_i}$) doit avoir une des formes suivantes :

- $expr_{E_i} = pred$
- $expr_{E_i} =$ `not` $(pred)$

où $pred$ est un prédicat. Nous supposons aussi que E et F ne sont pas équivalents à $vrai$. La définition suivante introduit la corrélation qui se base sur la notion d'unification de la logique du premier ordre.

Définition 5 : Corrélation Les expressions E et F sont *corrélées* s'il existe i dans $\{1, .., m\}$ et j dans $\{1, .., n\}$ tels que $expr_{E_i}$ et $expr_{F_j}$ sont unifiables par un plus grand unifieur (mgu) Θ.

Définition 6 : Corrélation d'actions ou influence positive Une action A a une *influence positive* sur une action B si la post condition de A et la pré-condition de B sont corrélées en utilisant la définition 4.

La figure 4 illustre un exemple de corrélation d'attaques où l'action *mount* peut avoir une influence positive sur l'action *.rhost modification*.

[1] Nous considérons que ces deux expressions n'incluent pas de disjonction. Cette restriction nous permet de simplifier la définition de la corrélation. La généralisation de la notion de corrélation afin de considérer les disjonctions représente un travail qu'il reste à faire.

Définition 7 : Action malveillante Une action A est une action malveillante pour l'objectif d'intrusion O si la post condition de A et la condition sur l'état du système de O sont corrélées.

En appliquant la définition 7, l'action $get\text{-}file(Agent, File, Printer)$ est une action malveillante car sa post condition est corrélée avec l'objectif d'intrusion $illegal\_file\_access(File)$.

Définition 8 : Action initiale Une action A est une action initiale si sa pré condition est égale à vrai ou si tous les prédicats de la pré condition sont satisfaits par l'état du système.

En appliquant la définition 8, l'action $touch(Agent, File)$ est une action initiale car sa pré condition est égale à vrai.

IV.5. Scénario d'intrusion

Un scénario d'intrusion est une séquence d'actions qui a pour but d'atteindre un certain état du système dans lequel un objectif d'intrusion est réalisé.

Définition 9 : Scénario d'intrusion Un scénario d'intrusion est une séquence $S(A_1, A_2, ..., A_n, O)$ ou les A_i sont des instances d'actions et O est une instance d'objectif d'intrusion telles que :

- $\forall i, j \in \{1, .., n\}$, si $i > j$ alors $Detectime(A_i) \geq Detectime(A_j)$.
- $\forall i \in \{2, .., n\}$, $\exists j < i$ tel que A_j a une influence positive sur A_i.
- A_n est une action malveillante pour O.

Il est à noter que d'autres actions (différentes de A_n) du scénario S peuvent aussi avoir une influence positive sur O. Un scénario peut comporter plusieurs actions initiales.

V. Exemple et limites

La définition de la corrélation donnée dans la section 4 est assez faible. En effet deux actions sont corrélées dès qu'elles ont un prédicat en commun dans leur pré et post conditions. Ainsi étant donné un ensemble d'actions, nous pouvons construire un nombre élevé de scénarios qui débouchent sur un même objectif d'intrusion. Afin d'illustrer ce point, nous prenons l'exemple d'un scénario aboutissant à un accès illégal à un fichier protégé.

Considérons un intrus, bad_guy, et un fichier confidentiel $secret\_file$. Supposons que bad_guy veuille atteindre l'objectif d'intrusion $illegal\_file\_access(secret\_file)$. Le système est au départ dans l'état suivant :

- $file(secret\_file)$
- not$(read_access(bad_guy, secret_file))$
- $printer(ppt)$
- $physical\_access(bad\_guy, ppt)$

– not($blocked(ppt)$)

Ce qui signifie que $secret\_file$ est un fichier, bad_guy n'a pas les droits pour le lire et ppt est une imprimante non bloquée à laquelle bad_guy a un accès physique. bad_guy veut atteindre un état du système dans lequel les prédicats suivants sont vrais :

– $read\_access(bad\_guy, secret\_file))$

– not($authorized((bad\_guy, read, secret\_file))$

C'est-à-dire bad_guy peut lire le fichier confidentiel $secret\_file$ alors qu'il n'en a pas le droit.

Supposons que les instances d'actions suivantes sont détectées :

– $A = touch(bad\_guy, guy\_file)$ avec $Detectime(A) = t_1$

– $B = block(bad\_guy, ppt)$ avec $Detectime(B) = t_2$

– $C = lpr\text{-}s(bad\_guy, ppt, guy\_file)$ avec $Detectime(C) = t_3$

– $D = remove(bad\_guy, guy\_file)$ avec $Detectime(D) = t_4$

– $E = ln\text{-}s(bad\_guy, guy\_file, secret\_file)$ avec $Detectime(E) = t_5$

– $F = unblock(bad\_guy, ppt)$ avec $Detectime(F) = t_6$

– $G = print\text{-}process(ppt, guy\_file)$ avec $Detectime(G) = t_7$

– $H = get\text{-}file(bad\_guy, secret\_file)$ avec $Detectime(H) = t_8$

Les dates des actions sont telles que $t_1 < t_2 < t_3 < t_4 < t_5 < t_6 < t_7 < t_8$.

V.1. Scénarios possibles

A partir de l'ensemble des actions instanciées présentées ci-dessous et à partir de la définition de la corrélation d'actions, nous pouvons construire plusieurs scénarios plausibles qui sont tous corrélés avec l'objectif d'intrusion constitué par l'accès illégal à un fichier. La figure 5 montre les liens de corrélation existants dans notre exemple et l'ordre chronologique de leur exécution.

V.1.1. Analyse hors ligne

Nous avons dit précédemment que la corrélation d'actions peut être utilisée dans deux approches différentes. La première approche consiste à considérer l'ensemble des 8 instances d'actions de l'exemple et à générer l'ensemble des scénarios expliquant ces actions. Une fois l'ensemble des scénarios possibles générés, il nous reste à choisir, dans cet ensemble, le scénario le plus plausible.

De la figure 5 nous pouvons déduire l'ensemble des scénarios possibles à partir des 8 actions observées :

– scénario 1 : $S_1 = (A, B, C, D, E, F, G, H, O)$

– scénario 2 : $S_2 = (A, C, G, H, O)$

– scénario 3 : $S_3 = (A, D, E, G, H, O)$

– scénario 4 : $S_4 = (B, F, G, H, O)$

– scénario 5 : $S_5 = (A, C, D, E, G, H, O)$

– scénario 6 : $S_6 = (A, B, D, E, F, G, H, O)$

– scénario 7 : $S_7 = (A, B, C, F, G, H, O)$

Il est à noter que seul le scénario 1 fait intervenir toutes les actions instanciées $(A - H)$. A et B sont des actions initiales car $Pre(A)$ est vraie et tous les

FIG. 5 – Graphe de corrélation du scénario d'accès non autorisé à un fichier

prédicats de $Pre(B)$ sont satisfaits par l'état initial du système.

Un administrateur système parviendrait à la conclusion que le scénario le plus dangereux parmi les 7 scénarios possibles est le premier, qui utilise toutes les actions détectées. Cela ne veut pas dire que le scénario maximal faisant intervenir toutes les instances d'action est le plus plausible mais dans notre exemple c'est celui qui a le plus de chances de réussir. En effet, le scénario 1 est le scénario où toutes les pré-conditions des actions sont satisfaites ou corrélées. Par exemple, pour l'action G du scénario 1, toutes les pré-conditions sont corrélées, alors que dans le scénario 4, un seul prédicat est corrélé. Nous voudrions être capable de choisir automatiquement le scénario le plus plausible parmi les 7 générés. De manière plus générale, étant donné un ensemble d'instances d'actions, nous voudrions être capables de générer un ensemble de scénarios possibles et de choisir le plus plausible parmi eux. Ce traitement des alertes de détection d'intrusions s'effectue donc après que l'intrus ait terminé toutes ses actions.

V.1.2 Analyse on-line

La deuxième approche utilisant notre définition de la corrélation consiste à considérer les instances d'action au fur et à mesure qu'elles sont générées. Lorsque une nouvelle alerte arrive, le but de la corrélation est de trouver l'ensemble des actions pouvant être exécutées après l'action observée afin d'anticiper les intentions de l'attaquant. La corrélation nous permet de trouver pas seulement les actions pouvant être exécutées juste après l'observation, mais aussi l'ensemble des actions pouvant être exécutées après n actions suivant l'observation. Pour des questions de complexité il est raisonnable de fixer un nombre maximal d'étapes anticipées dans le scénario. Une fois que l'ensemble des scénarios complets ou partiels sont générés après une observation, il reste à déterminer lequel de ces scénarios ou début de scénarios est le plus plausible et donc celui à surveiller.

Afin de résoudre le problème du choix du scénario le plus plausible dans les deux approches citées, nous introduisons dans la prochaine section deux améliorations que sont les notions d'anti-corrélation et de corrélation pondérée.

VI. Corrélation pondérée

La première idée pour discerner les scénarios possibles consiste à introduire la notion d'anti-corrélation ([13]), ou encore d'influence négative, entre deux actions. Intuitivement, A anti-corrèle B si lorsque A est exécutée, B ne peut pas être immédiatement exécutée. Plus précisément, A anti-corrèle B s'il existe une expression $expr_1$ dans $Post(A)$ et une expression $expr_2$ dans $Pre(B)$ telles que $expr_1$ et $\texttt{not}(expr_2)$ sont unifiables.

Définition 10 : Anti-correlation Deux expressions logiques E et F sont anti-corrélées si une des conditions suivantes est satisfaite :

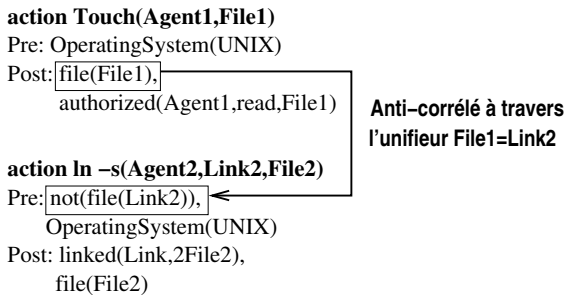

FIG. 6 – Exemple d'anti-corrélation entre deux actions

- il existe i dans $[1, m]$ et j dans $[1, n]$ tels que $expr_{E_i}$ et `not` $(expr_{F_j})$ sont unifiables via un mgu Θ.
- il existe i dans $[1, m]$ et j dans $[1, n]$ tels que `not` $(expr_{E_i})$ et $expr_{F_j}$ sont unifiables via un mgu Θ.

Définition 11 : Influence négative Une action A a une *influence négative* sur une action B si $Pre(A)$ et $Post(B)$ sont anti-corrélées d'après la définition 10.

Par exemple, la post condition de $touch(Agent1, File1)$ est anti-corrélée avec la pré-condition de $ln\text{-}s(Agent2, Link2, File2)$ via l'unifieur $File1 = Link2$ (voir figure 6).

L'anti-corrélation nous permet d'ignorer des scénarios contenant au moins une action ayant une influence négative sur une autre action. Ceci est notamment intéressant lors de l'analyse en temps réel des alertes car l'élimination de certains scénarios diminue les temps de calcul.

La seconde amélioration que nous proposons est d'associer avec chaque instance d'action B, dans un scénario donné, un poids de corrélation. Ce poids dépend de l'ensemble des actions ayant une influence, positive ou négative, sur B.

Dans la suite nous désignons par S_B l'ensemble des actions appartenant au scénario S et qui ont une influence sur B. Le poids de corrélation associé à B est allors défini à partir du nombre de prédicats de $Pre(B)$ qui peuvent être unifiés avec les post conditions des actions de l'ensemble S_B. Plus formellement soit :

- $Pos(S_B) = \bigcup_{A \in S_B} Post(A)$
- $U(S_B, B)$: le nombre de prédicats dans $Pre(B)$ qui sont unifiés avec au moins un élément de $Pos(S_B)$

Alors :

Définition 12 : Poids de corrélation le poids de corrélation associé à une action B dans un scénario S, noté par $\omega_S(B)$, est défini de la manière suivante :

$$\omega_S(B) = \begin{cases} 0 & \text{s'il existe au moins un élément dans } S_B \\ & \text{ayant une influence négative sur } B \\ 1 & \text{si } S_B = \emptyset \text{ (c'est-à-dire B est une action initiale)} \\ \frac{U(S_B,B)}{|Pre(B)|} & \text{sinon} \end{cases}$$

VII. Pondération des scénarios

Cette section montre comment utiliser les poids de corrélation afin d'induire une relation d'ordre sur un ensemble de scénarios conduisant au même objectif d'intrusion.

VII.1. Approche hors-ligne

Dans la suite, à chaque scénario $S = (A_1, A_2, ..., A_n, O)$ nous associons son vecteur de poids $(\omega_S(A_1), \omega_S(A_2), ..., \omega_S(A_n), \omega_S(O))$. Les questions qui se posent sont comment agréger ces poids afin d'évaluer la plausibilité d'un scénario donné et comment comparer deux scénarios pondérés. Nous notons g l'opérateur d'agrégation.

Le premier mode d'agrégation le plus naturel est de considérer l'opérateur moyenne, c'est-à-dire :

Mode d'agrégation par la moyenne :

$$g(A_1, ..., A_n, O) = \frac{\sum_{i=1}^{n} \omega_S(A_i) + \omega_S(O)}{n+1}$$

Cependant ce mode d'agrégation est indésirable car des scénarios ayant des poids très différents pour chaque action les composant pourraient être considérés comme étant également plausibles.

Mode d'agrégation conjonctif :

Une condition naturelle que g doit satisfaire est : si $\exists i \in \{1, .., n\}$ $\omega_S(A_i) = 0$ ou $\omega_S(O) = 0$ alors $g(A_1, ..., A_n, O) = 0$. Les fonctions d'agrégation satisfaisant cette condition sont appelées les opérateurs conjonctifs. Une forme plus faible de tels opérateurs peut être que si $\omega_S(A_i) = 0$ ou $\omega_S(O) = 0$ alors le scénario S devrait être parmi les scénarios les moins plausibles. La forme la plus faible d'un opérateur conjonctif serait de dire qu'un scénario S ne doit pas être parmi les scénarios les plus plausibles si $\omega_S(A_i) = 0$ ou $\omega_S(O) = 0$.

Un exemple d'opérateur d'agrégation conjonctif est l'opérateur minimum :

Définition 13 : Un scénario $S = (A_1, A_2, ..., A_n, O)$ est dit plus plausible que $S' = (B_1, B_2, ..., B_{n'}, O)$ si
$min((\omega_S(A_1), ..., \omega_S(A_n), \omega_S(O))) > min((\omega_{S'}(B_1), ..., \omega_{S'}(B_{n'}), \omega_{S'}(O)))$

Cette définition considère qu'un scénario est plausible dès lors que le plus petit poids de corrélation d'une action est non nul. Plus le poids est élevé, plus le scénario est plausible.

Cette définition est cependant trop restrictive. Supposons que nous ayons deux scénarios $S = (A_1, A_2, ..., A_n, O)$ et $S' = (B_1, B_2, ..., B_{n'}, O)$. Supposons que le scénario S est tel que $\forall i \in [1, n], \omega_S(A_i) = \alpha$ et $\omega_S(O) = \alpha$. Supposons que le scénario S' est tel que $\exists j, \omega_{S'}(B_j) = \alpha$ et que $\forall i \in [1, n'], i \neq j, \omega_{S'}(B_i) > \alpha$ et $\omega_{S'}(O) > \alpha$. Dans un tel cas il est clair que l'on préfère le scénario S' au scénario S étant donné que S' contient des actions plus fortement corrélées. Mais si l'on considère l'opérateur minimum ces deux scénarios ont la même plausibilité car on ne considère que l'action la moins corrélée.

Un raffinement possible de l'opérateur minimum est d'utiliser l'opérateur dit lexicographe, bien connu dans la théorie du choix social [1]. Cet opérateur n'est défini que lorsqu'il est appliqué à deux vecteurs de même taille. En conséquence lors de la comparaison de deux scénarios ne comportant pas le même nombre d'actions, nous devons dupliquer le poids le plus faible du scénario le plus court afin d'obtenir deux vecteurs de taille identique. La définition suivante définit l'opérateur "leximin" :

Définition 14 : Soient $\overrightarrow{v} = (v_1, ..., v_n)$ et $\overrightarrow{v}' = (v'_1, ..., v'_n)$ deux vecteurs de poids ordonnés dans l'ordre croissant des poids, c'est-à-dire $v_1 > ... > v_n$ et $v'_1 > ... > v'_n$. Alors $\overrightarrow{v}$ est dit "leximin" préféré à $\overrightarrow{v}'$, noté $\overrightarrow{v} >_{leximin} \overrightarrow{v}'$, si $\exists i$ tel que $v_i > v'_i$ et $\forall j < i$, $v_j = v'_j$.

Afin d'appliquer cette définition pour ordonner des scénarios, nous représentons les poids de corrélation associés à chaque scénario comme un vecteur de poids $\overrightarrow{v_S}$. Par $\overrightarrow{v_{\sigma(S)}}$ nous désignons le vecteur obtenu à partir de $\overrightarrow{v_S}$ en ordonnant les poids dans l'ordre croissant.

Dès lors, le choix de scénarios plausibles est donné par la définition suivante :

Définition 15 : Un scénario S est préféré à S', noté $S > S'$, si $\overrightarrow{v_{\sigma(S)}} >_{leximin} \overrightarrow{v_{\sigma(S')}}$. Un scénario S est parmi les scénarios les plus plausibles s'il n'y a pas de scénario S' tel que $S' > S$.

Remarquons que cette définition ne favorise pas systématiquement les scénarios contenant le plus d'actions. En effet, considérons deux scénarios, le premier contenant plus d'actions que le deuxième. Le premier scénario peut cependant être écarté s'il contient, par exemple, une relation d'anti-correlation. Retournons à notre exemple d'accès illégal à un fichier. Comme il a été dit dans la section 5.1, d'après la définition 6 nous pouvons construire les 7 scénarios suivants :

– scénario 1 : $S_1 = (A, B, C, D, E, F, G, H, O)$

– scénario 2 : $S_2 = (A, C, G, H, O)$
– scénario 3 : $S_3 = (A, D, E, G, H, O)$
– scénario 4 : $S_4 = (B, F, G, H, O)$
– scénario 5 : $S_5 = (A, C, D, E, G, H, O)$
– scénario 6 : $S_6 = (A, B, D, E, F, G, H, O)$
– scénario 7 : $S_7 = (A, B, C, F, G, H, O)$

D'après la définition 12, leur vecteurs de poids correspondants sont :
– scénario 1 : $\overrightarrow{v_{S_1}} = (1, 1, 1, 1, 1, \frac{1}{2}, 1, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_1)}} = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1, 1, 1, 1)$
– scénario 2 : $\overrightarrow{v_{S_2}} = (1, 1, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_2)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, 1, 1)$
– scénario 3 : $\overrightarrow{v_{S_3}} = (1, 1, 1, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_3)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1)$
– scénario 4 : $\overrightarrow{v_{S_4}} = (1, \frac{1}{2}, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_4)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1)$
– scénario 5 : $\overrightarrow{v_{S_5}} = (1, 1, 1, 1, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_5)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1, 1)$
– scénario 6 : $\overrightarrow{v_{S_6}} = (1, 1, 1, 1, \frac{1}{2}, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_6)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1, 1)$
– scénario 7 : $\overrightarrow{v_{S_7}} = (1, 1, 1, \frac{1}{2}, \frac{1}{3}, \frac{1}{2}, \frac{1}{2})$ et $\overrightarrow{v_{\sigma(S_7)}} = (\frac{1}{3}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, 1, 1)$

D'après la définition 14, les 7 scénarios sont ordonnés de la manière suivante : $S_1 > S_6 > S_5 > S_7 > S_3 > S_2 > S_4$

Donc le scénario S_1, qui fait intervenir toutes les actions instanciées, est le scénario retenu.

VII.2. Approche en ligne

L'approche en ligne diffère peu de l'approche hors ligne excepté le fait que nous ne comparons pas des scénarios nécessairement complets. Par scénario incomplet nous désignons un scénario ne comportant pas encore d'objectif d'intrusion. Un tel scénario est généré par l'algorithme de corrélation qui, à partir des observations (les alertes), génère des suites d'actions compatibles avec les observations. Le nombre maximum d'actions anticipées par l'algorithme est limité pour des raisons de coût de calcul. Par exemple considérons le cas de notre scénario d'accès illégal à un fichier. Une fois que nous avons reçu la première alerte correspondant à l'action *Touch*, si nous fixons à 1 le nombre limite d'actions anticipées, trois scénarios sont générés. Le premier est composé de *Touch* et *lpr -s*, le deuxième de *Touch* et *Remove* et le troisième de *Touch*, *lpr -s* et *Remove*, *lpr -s* et *Remove* étant tous deux corrélés à *Touch*. Pour dégager le scénario le plus fortement corrélé, il nous suffit d'appliquer les définitions précédentes en omettant dans le calcul du vecteur de poids le poids associé à l'objectif s'il n'est pas présent dans le scénario généré.

VIII. Conclusion

Nous nous sommes basés sur le fait qu'un scénario d'intrusion peut être représenté par un processus de planification. Nous avons proposé un modèle permettant de reconnaître des scénarios d'intrusion et les intentions malveillantes associées. Ce modèle se démarque des modèles précédent (par exemple [8, 9]) qui nécessitent de spécifier une librairie de scénarios d'intrusion. Notre approche est basée sur la spécification d'actions élémentaires et d'objectifs d'intrusion. Nous

avons ensuite montré comment dériver des relations de corrélation, ou d'influence positive, entre deux instances d'action ou entre une instance d'action et une instance d'objectif d'intrusion. La détection d'intrusions complexes se fait en combinant ces relations binaires de corrélation. Nous avons ensuite défini la notion d'anti-corrélation, ou d'influence négative, qui est utile pour identifier une séquence d'actions corrélées qui ne permettent plus à l'attaquant d'atteindre un objectif d'intrusion. Cela peut être utilisé pour éliminer une catégorie de faux positifs qui correspondent à de fausses attaques, c'est-à-dire des actions qui ne sont plus corrélées à un objectif d'intrusion. Enfin, nous avons présenté la corrélation pondérée qui peut être utile pour choisir les scénarios les plus plausibles. Quand l'attaquant n'a pas encore atteint son objectif d'intrusion et qu'il existe plusieurs objectifs d'intrusion corrélés avec une séquence donnée d'action, notre stratégie actuelle consiste à choisir l'objectif le plus fortement corrélé.

Un travail futur serait d'exploiter la notion de corrélation pondérée dans le cas où des actions peuvent ne pas être détectées. Dans [12] une solution est proposée basée sur la génération d'alertes virtuelles. Le nombre d'alertes virtuelles peut être important, et l'utilisation de la pondération peut limiter le nombre d'hypothèses possibles à celles qui sont les plus plausibles.

Un autre travail futur est de voir comment intégrer une connaissance experte dans le processus de corrélation. Pour décider si une instance de scénario est achevée ou non, il est souvent nécessaire de combiner les informations provenant des SDIs avec d'autres informations sur le système surveillé : sa topologie, sa configuration ainsi que d'autres données sur le type et la version du système d'exploitation et sur les applications installées [2]. Ce type de données n'est pas fourni par les SDI classiques mais il existe d'autres outils qui peuvent être utilisés pour les collecter ([11]). Étant donné que les SDI actuels ne nous fournissent pas de données nous permettant de savoir si une action est un succès ou un échec, ces données additionnelles peuvent être très utiles.

Remerciements

Ce travail à été financé par le ministère Français de la recherche dans le cadre du projet RNTL DICO. Les auteurs remercient tous les membres de ce projet et tout particulièrement les autres membres du sous-projet "corrélation" : Hervé Debar, Ludovic Mé et Benjamin Morin.

Références

[1] Moulin (H.), Axioms of Cooperative Decision Making, *Cambridge University Press*, Cambridge, 1988.

[2] Huang (M.), A Large-scale Distributed Intrusion Detection Framework Based on Attack Strategy Analysis, *Proceedings of the First Internationnal Workshop on the Recent Advances in Intrusion Detection (RAID'98)*, Louvain-La-Neuve, Belgium, 1998.

[3] Mé (L.), Marrakchi (Z.), Michel (C.), Debar (H.) and Cuppens (F.), La détection d'intrusion : les outils doivent coopérer. REE journal.

[4] Cuppens (F.) and Ortalo (R.), Lambda : A language to model a database for detection of attacks, *Third International Workshop on the Recent Advances in Intrusion Detection (RAID'2000)*, Toulouse, France, October 2000.

[5] Valdes (A.) and Skinner (K.), Probabilistic Alert Correlation, *Fourth International Workshop on the Recent Advances in Intrusion Detection (RAID'2001)*, Davis, USA, October 2001.

[6] Dain (O.) and Cunningham (R.), Building Scenarios from a Heterogeneous Alert Stream, *proceedings of the 2001 IEEE, Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, 5-6 June 2001

[7] Debar (H.) and Wespi (A.), The Intrusion Detection Console Correlation Mechanism, *Workshop on the Recent Advances in Intrusion Detection (RAID'2001)*, Davis, USA, October 2001.

[8] Geib (C.) and Goldman (R.), Plan Recognition in Intrusion Detection Systems, *DARPA Information Survivability Conference and Exposition (DISCEX)*, June 2001.

[9] Geib (C.) and Goldman (R.), Probabilistic Plan Recognition for Hostile Agents, *Florida AI Research Symposium (FLAIR)*, Key-West, USA, 2001.

[10] Cuppens (F.), Managing alerts in a multi-intrusion detection environment, *17th Annual Computer Security Applications Conference*, New Orleans, Louisiana, December 10-14, 2001.

[11] Morin (B.), Mé (L.), Debar (H.), Ducassé (M.), M2D2 : A Formal Data Model for IDS Alert Correlation, *Recent Advances in Intrusion Detection, 5th International Symposium, RAID 2002*, Zurich, Switzerland, October 16-18, 2002

[12] Cuppens (F.) and Miège (A.), Alert Correlation in a Cooperative Intrusion Detection Framework, *IEEE Symposium on Security and Privacy*, Oakland, USA, 2002.

[13] Cuppens (F.), Autrel (F.), Miège (A.), Benferhat (S.), Recognizing malicious intention in an intrusion detection process, *Second International Conference on Hybrid Intelligent Systems (HIS'2002)*, Santiago, Chile, October 2002.

[14] Sheyner (O.), Haines (J.), Jha (S.), Lippmann (R.) and Wing (J.), Automated Generation and Analysis of Attack Graphs, *Proceedings of IEEE Symposium on Security and Privacy*, May 2002.

[15] Ning (P.), Cui (Y.) and Reeves (D.), Constructing Attack Scenarios Through Correlation of Intrusion Alerts, *proceedings of ACM CCS 02*

[16] Ning (P.), Xu (D.), Learning Attack Strategies from Intrusion Alerts, *proceedings of ACM CCS 03*

[17] Debar (H.), Morin (B.), Cuppens (F.), Autrel (F.), Mé (L.), Vivinis (B.), Benferhat (S.), Ducassé (M.), Ortalo (R.), Détection d'intrusions : corrélation d'alertes, em Revue TSI 23/2004. Sécurité informatique, pages 359 à 390