Classification des Attaques pour l'évaluation des IDS

A. Abou El Kalam[α], M. Gad El Rab[β] et Y. Deswarte[β]
[α] *LIFO - ENSI de Bourges, 88 Bd Lahitolle, 18000 Bourges, France.*
E-Mail : anas.abouelkalam@ensi-bourges.fr
[β] *LAAS – CNRS, Université de Toulouse, France*
E-Mail: {gad-el-rab, deswarte}@laas.fr

Confrontés à une multiplication et à une complexité croissante des attaques, les évaluateurs des systèmes de détection d'intrusion ont besoin d'organiser, de structurer et de classifier les attaques pour mieux choisir les jeux de test. En effet, il serait impossible de générer tous les cas de test possible en tenant compte de toutes les attaques éventuelles (connues et inconnues !).

Pour pallier ce type de problèmes, nous proposons une nouvelle méthode pour la classification des attaques. Ensuite, nous utilisons cette classification pour présenter une approche simple de sélection des cas de test. Pour cela, nous proposons d'utiliser une méthode basée sur l'arbre de classification (CTM, pour *Classification Tree Method*).

La combinaison de notre classification, de la méthode et de la prise en compte des contraintes du système permet de réduire considérablement les cas de test, tout en ayant une idée claire sur la robustesse de l'IDS vis-à-vis de tel ou tel type d'attaques.

Enfin, nous utilisons l'outil CTE (*Classification Tree Editor*) pour générer et sélectionner les cas de test.

Mots-clés: Systèmes de Détection d'Intrusions, classification d'attaques, sélection des cas de test.

1. Introduction

Pour améliorer la sécurité des réseaux, les administrateurs disposent de nombreux outils, dont les systèmes de détection d'intrusions (IDS pour *Intrusion Detection Systems* en anglais). Ces outils ont connu un essor particulier au cours des dernières années, notamment en raison du nombre grandissant d'attaques.

Néanmoins, ce même argument qui a permis un déploiement massif des IDS, pose de sérieux problèmes pour les évaluateurs de tels systèmes. En effet, comment tester efficacement et avoir la certitude (prouver) que l'IDS se comporte correctement (par ex. génération d'alarme lors d'une tentative d'intrusion, pas de fausses alertes, etc.) pour toutes les attaques existantes ?

Une solution qui peut paraître triviale consiste à construire des classifications pertinentes et représentatives de toutes les attaques. L'idée est de réduire considérablement les cas possibles en construisant des classes d'attaques de telle manière que le test ne prendra qu'un élément de chaque classe. Cette technique part du principe que n'importe quelle instance d'attaque d'une classe donnée produira les mêmes effets, et donc génère les mêmes résultats.

Dans cette logique, plusieurs travaux ont essayé de "cataloguer" les vulnérabilités (par ex. dans des bases de données), dans le but de faciliter l'analyse et l'identification des caractéristiques communes des vulnérabilités connues [4, 15]. De la même manière, certains ont proposé de répertorier et classifier les attaques pour aider à gérer les incidents et pour traiter les informations d'audit [6, 9, 11, 12]. Dans la majorité des cas, les classifications existantes utilisent les informations et les points de vues du CERT (pour *Computer Emergency and Response Team*).

Pourtant, si les classifications de vulnérabilités (par ex. CVE, OSVDB [4, 15, 20, 21]) sont maintenant largement supportées et implémentées dans plusieurs produits et outils de sécurité, aucune

classification d'attaques n'est reconnue comme un standard, ni utilisée à grande échelle. De plus, les classifications existantes s'avèrent peu utiles pour l'évaluation des IDS (elles sont mal adaptées à ce type de traitement).

En réalité, le problème qui nous intéresse − l'évaluation des IDS − souffre de trois problèmes majeurs :

1. l'absence de méthode systématique pour l'évaluation ;
2. l'utilisation de métriques biaisées, qui négligent la sensibilité et la complexité des données de test ; et
3. le manque de données réalistes qui peuvent être utilisées pour le test ou pour modéliser les problèmes à résoudre et valider les expériences.

Nous avons étudié les deux premiers points dans [17] ; dans cet article, nous nous intéressons au troisième point, la non-représentativité des données de test.

Pour traiter ce problème, il nous semble nécessaire de commencer par une analyse approfondie des classifications existantes d'attaques et de vulnérabilités, et de mettre ainsi en évidence leurs forces et faiblesses. Ensuite, il serait judicieux d'étudier les attributs de chaque taxonomie afin d'extraire ce qui peut être pertinent pour notre étude (l'évaluation des IDS). En effet, chacune des classifications existantes a été développée dans un but particulier (par ex., comprendre les vulnérabilités pour renforcer les mesures correctives et défensives, appréhender les processus d'attaque ainsi que le comportement des attaquants, etc.) ; les attributs identifiés dans une étude ne sont donc pas forcément pertinents pour une autre ayant un objectif différent.

Nous éliminerons ainsi les attributs les moins utiles et nous en proposerons d'autres ; le but étant d'aboutir à une classification adaptée à l'évaluation des IDS.

Par ailleurs, afin d'arriver à une sélection "semi-automatique" des jeux de test, nous proposerons d'appliquer à notre classification une méthode basée sur l'arbre de classification.

Ce travail peut ainsi servir à plusieurs fins, notamment :

- réduire le nombre d'attaques utilisées dans un jeu de test, afin d'en faciliter la gestion ;
- mieux couvrir l'espace (domaine) des attaques et savoir avec précision contre quelles attaques l'IDS s'est correctement comporté (et par induction, contre quelles attaques l'IDS se comportera correctement) ;
- avoir une meilleure connaissance de l'espace des attaques et améliorer ainsi notre compréhension des instances d'attaques, y compris les plus récentes, mais aussi celles qui sont envisageables ou à venir.

Avant d'entrer dans le vif du sujet et afin d'éviter toute confusion, nous présentons tout d'abord quelques termes et définitions utiles.

Le terme attaque désigne toute action malveillante qui essaye de violer la politique de sécurité ; une intrusion est toute attaque qui a réussi à exploiter une vulnérabilité (faille ou faiblesse) et donc à pénétrer illégalement dans le système ou de l'empêcher de remplir le service attendu (ex., attaques par déni de service).

Dans le contexte de la détection d'intrusion, le terme "classification" est souvent utilisé pour exprimer une "distinction" ou "identification" ; mais il est parfois utilisé pour exprimer des priorités, par exemple, selon le niveau de criticité de la menace, Snort classifie les attaques en : *bas*, *moyen* ou *élevé* [24]. Dans ce travail, les termes de taxonomie et de classification seront utilisés dans le sens de *catégorisation*.

Enfin, les termes attributs, axes et dimensions seront utilisés pour préciser les critères ou les caractéristiques de la classification ou taxonomie.

Le reste de cet article est organisé comme suit : la section 2 présente les différentes taxonomies d'attaques existantes. La section 3 vient ensuite dresser une analyse critique de la pertinence de ces travaux et discuter leur adaptabilité pour la sélection des attaques pour le test des IDS. Dans les Sections 4 et 5 nous introduisons respectivement notre classification ainsi que d'autres approches complémentaires destinées à la sélection des attaques pour évaluer les IDS. Enfin, la Section 6 conclut ce travail et en présente les perspectives.

2. Analyse des classifications existantes

Le but de cette section est de dresser un état de l'art des taxonomies existantes, mais surtout de voir (*cf.* section suivante) si elles peuvent être pertinentes pour l'évaluation des IDS. Une description plus générale des différentes taxonomies peut être trouvée dans [12].

Commençons par la taxonomie de Bishop [2] ; même si celle-ci concerne les vulnérabilités, il peut être intéressant de regarder les attributs (axes) qu'elle considère : *nature de la faille* (ex. débordement de tampon), *phase de l'introduction de la vulnérabilité* (ex., pendant l'étape de conception ou d'implémentation), *domaine d'exploitation* (c'est-à-dire comment l'exploiter), *domaine des effets* (c-à-d ce qui est affecté), *nombre minimum des composants* nécessaires à l'exploitation de cette vulnérabilité et *source* de son identification (c-à-d le site ou la liste de diffusion où la vulnérabilité a été publiée).

Kumar a proposé une classification des attaques selon quatre attributs du schéma ou de la signature de l'attaque : *existence*, *séquence*, *intervalle* et *durée* [9].

La taxonomie de Hansman considère quatre dimensions reliée aux attaques : ce qu'il appelle le *vecteur* ou le type (c-à-d le moyen utilisé par l'attaquant pour arriver à ses fins, comme les virus, les vers, le déni de service), la *cible* (ex. système d'exploitation, protocole réseau), les *effets* de l'attaque ainsi que la *vulnérabilité exploitée* [18].

Un autre travail de base est celui de Lindqvist et Jonsson [11] qui étendent la taxonomie de Newman et Parker [23]. Ces derniers considèrent une seule dimension, la *technique*, tandis que les autres ajoutent le *résultat* comme dimension. Cette classification s'inscrivait dans le cadre d'expériences menées par des utilisateurs internes (étudiants d'une classe d'informatique) afin d'améliorer les capacités de détection d'IDS qui utilisent le filtrage par reconnaissance des formes (*pattern matching*, en anglais). D'ores et déjà, on remarque que ce travail ne considère que des attaques lancées par des étudiants. On peut ainsi constater qu'elle ignore une grande partie de l'espace des attaques, notamment des attaques plus sophistiquées (non imaginées ou non accessibles aux étudiant de cette classe).

Weber a présenté une taxonomie basée sur trois dimensions : le *niveau de privilège requis* pour mener l'attaque, le *moyen utilisé par l'attaquant* (ex. exploitation d'un bug logiciel) ainsi que *l'effet souhaité* (ex. déni de service) [19].

La taxonomie de la DARPA est en fait une version réduite de celle de Weber. Elle ne considère que *l'effet* de l'attaque comme dimension. Les attaques sont divisées en cinq catégories : "distant vers local" (ou R2L pour *Remote to Local*), "utilisateur vers super-utilisateur" (ou U2R pour *User to Root*), "sonder" (*scan*) et "déni de service" [7, 11, 22]. Là encore, on peut remarquer que cette classification considère des niveaux différents d'abstractions, ce qui pose des problèmes, notamment l'exclusion mutuelle des classes résultantes.

À la différence des classifications déjà citées, la taxonomie de Howard est centrée sur le processus de l'attaque, plutôt que sur l'attaque elle même [6]. Elle tient compte de l'*attaquant* (qui est-il ?), de l'*outil* qu'il a utilisé, de la *vulnérabilité* exploitée, de *l'accès obtenu*, des *résultats* de l'attaque (c-à-d divulgation, altération) ainsi que ses *objectifs* (c-à-d obtenir ou détruire une information).

Un autre travail intéressant présente une taxonomie prenant un point de vue défensif [8]. Le but était de fournir des informations pour aider les administrateurs à mieux défendre leurs systèmes. Les attaques ont ainsi été classifiées selon leurs manifestations (opérations visibles) telles qu'elles sont vues par des HIDS (*Anomaly Host-based IDS*, en anglais). Les quatre dimensions de cette taxonomie sont :

1- *signes extérieurs* : il s'agit d'appels "systèmes" qui apparaissent suite à l'exécution de l'attaque, mais qui n'apparaissent jamais dans les opérations normales (c-à-d dans les activités non-intrusives) ;

2- *séquence minimale* : c'est la plus petite séquence qui apparaît dans l'attaque, mais qui n'apparaît jamais dans des opérations normales ;

3- *séquence dormante* : c'est une séquence qui correspond (partiellement) à une sous-séquence d'opérations normales ;

4- *séquence normale* : c'est une séquence dans l'attaque qui ne se distingue pas des activités non-intrusives.

Enfin, regardons de plus près l'importante taxonomie d'Alessandri [1]. Celle-ci a été élaborée à des fins d'analyse des modèles des IDS. Au lieu de catégoriser directement les attaques, elle classifie plutôt toutes les activités (de manière plus globale) qui peuvent être pertinentes pour l'IDS. Une évaluation analytique a été ensuite établie pour déterminer les capacités de détection de l'IDS vis-à-vis de telle ou telle classe d'attaques.

Plus concrètement, le modèle correspondant à cette classification fait la différence entre les caractéristiques dynamiques et les caractéristiques statiques d'une activité observable par l'IDS. Les activités statiques sont divisées en caractéristiques reliées aux objets-interface[1] et en celles reliées aux objets affectés (corrompus) par l'attaque. Les caractéristiques dynamiques sont développées selon trois critères : caractéristiques de la communication (ex. unidirectionnelle, bidirectionnelle), méthode d'invocation (ex. création, suppression, lecture) ainsi que d'autres attributs additionnels qualifiés de mineurs (ex. l'attaque provient de plusieurs origines ou elle contient des événements répétitifs).

L'attaque, quant à elle, est décrite selon cinq paramètres : l'*objet-interface*, l'*objet affecté*, la *communication*, la *méthode d'invocation* ainsi que d'autres *attributs mineurs*. Au total, cette taxonomie contient 25 objets-interface, 10 objets affectés, 3 caractéristiques reliées à la communication, 5 méthodes d'invocations ainsi que 4 attributs additionnels mineurs.

Avant de présenter notre classification (dans la section 4), la section suivante va d'abord discuter les limites des classifications présentées et va analyser les attributs qu'elles proposent afin de ne retenir que les plus pertinents pour l'évaluation des IDS.

3. Discussion

Les différentes taxonomies existantes adoptent différents points de vue, et sont basées sur des attributs liés aux attaques ou aux vulnérabilités. Même s'il est impossible de citer dans cet article toutes les taxonomies existantes, on peut globalement identifier les attributs les plus importants :

- *type de l'attaque* : virus, vers, cheval de Troie, déni de service, etc. ;
- *technique de détection* de l'attaque : approche statistique, filtrage (*pattern matching*) ;
- *signature de l'attaque* : pattern ou séquence de patterns observés ;
- *outil utilisé par l'attaqu*ant : tool kit, script, commande utilisateur, etc. ;
- *cible de l'attaque* : système d'exploitation, protocole réseau, application, service, … ;
- *résultat de l'attaque* : modification illicite ou divulgation d'informations, déni de service, … ;
- *accès visé par l'attaque* : accès en super-utilisateur, accès en utilisateur normal ;
- *préconditions de l'attaque* : existence de versions particulières d'un certain logiciel, … ;
- *vulnérabilité exploitée par l'attaque* : débordement de mémoire, mauvaise stratégie de mot de passe, mauvaise configuration, etc. ;
- *objectif de l'attaque* : gain financier, terrorisme, autosatisfaction, etc. ;
- *localisation de l'attaque* : interne, externe ;
- *propriété de sécurité violée ou visée par l'attaque* : confidentialité, intégrité, disponibilité.

Certaines taxonomies utilisent un seul attribut ; ce type de travaux présente certaines faiblesses car les attaques d'une même classe sont disparates et n'ont ni un lien fort entre elles ni suffisamment de caractéristique communes.

Les taxonomies multidimensionnelles sont plus intéressantes car elles contiennent des classes dont les différences sont plus claires, et qui regroupent des attaques plus similaires. Néanmoins, ce type de taxonomies souffre du problème d'exclusion mutuelle. En effet, comment être sûr qu'aucune attaque ne puisse appartenir à deux classes différentes. Il est vrai que (comme le pensent certains auteurs) la nature même de certaines attaques (sophistiqués, composites) rend difficile le maintien de cette propriété (exclusion mutuelle). Néanmoins, on peut noter que la plupart des travaux existants

[1] Un objet interface est un objet (e.g., fichier, processus) qui contient une vulnérabilité ou qui propose une fonctionnalité qui a servie pour attaquer d'autres objets (entités du systèmes).

souffrent d'un manque de clarté dans la distinction entre les attributs, et donc entre les attaques. Par exemple, certaines classifications regroupent le débordement de tampon et le déni de service sous le même attribut ; cette exclusion mutuelle est abusive car une attaque qui exploite un débordement de tampon peut aussi causer un déni de service.

Par ailleurs, nous constatons que la plupart des classifications sont centrées sur l'attaquant, c-à-d adoptent la vision de l'attaquant (*attacker-centric*, en anglais). Ce type d'approches souvent ignore (ou masque) certaines caractéristiques importantes des attaques, telles qu'elles sont vues par l'IDS ou les administrateurs système.

À l'inverse, la classification d'Alessandri a été principalement créée pour l'analyse des modèles d'IDS [1]. Elle considère plus de détails reliés à l'attaque en terme de caractérisation des IDS, ce qui la rend plus pertinente pour l'évaluation et le test des IDS. Néanmoins, elle présente quelques limites. Tout d'abord, elle s'est centrée sur les manifestations des activités intrusives qui peuvent être observables par l'IDS, et elle ignore certains attributs intéressants pour l'évaluation des IDS, notamment les privilèges requis ou obtenus, les conséquences, etc.

De plus, nous trouvons son niveau de dimensionnement très fins, au point que le niveau de détail atteint n'est pas très utile pour le test des IDS. Pour prendre un exemple simple, la dimension "objet-interface", qui contient 24 types, considère les cinq types suivants, reliés à l'application : *App. layer-connectionless ; App. Layer single-connection single-transaction ; App. layer single-connection multiple-transaction ; App. layer multiple-connection single-transaction ;* et *App. layer-multiple-connection multiple-transaction*. Avec ce niveau fin de granularité, il n'est pas rare de trouver des classes contenant seulement une ou deux attaques.

Par ailleurs, les combinaisons des différents cas possibles (compte tenu de cette classification très fine) conduit à 9600 cas de test, alors que, par exemple, la fusion des classes reliées au niveau de l'application, permet de réduire ce nombre à 8000. On peut donc obtenir un gain considérable de temps, sans pour autant pénaliser la procédure de test. En effet, le niveau de détail caché lors du regroupement (proposé dans l'exemple) peut être investi ultérieurement à travers une analyse complémentaire, si l'IDS sous test est sensible à certains types de communication au niveau applicatif.

Pour résumer cette discussion, force est de constater que les taxonomies existantes ne sont pas réellement adaptées pour l'évaluation des IDS. Les raisons peuvent globalement être résumées dans les points suivants :

- dans leurs majorités, elles considèrent la vision de l'attaquant et pas celle de l'IDS ; il n'est donc pas étonnant que les attributs résultants soient moins pertinents pour le test des IDS ;
- parfois, la définition des attributs est quelque peu ambiguë voir incohérente ; des problèmes comme l'exclusion mutuelle s'imposent ainsi ;
- le nombre de classes résultantes est parfois très grand, sans que la complexité qui en résulte soit justifiée par une efficacité accrue du test des IDS ;
- ces classifications ne sont malheureusement pas accompagnées de schéma de sélection et génération des cas de test.

Dans le reste de cet article, nous proposons une nouvelle classification qui cherche à éviter ces limites.

Nous nous baserons sur les attributs que nous avons identifiés lors de l'étude des classifications existantes, en éliminant ceux qui sont ambigus ou qui ne sont pas pertinents pour l'évaluation des IDS. Les attributs retenus seront accompagnés par une définition claire. Enfin, nous allons combiner notre classification avec un schéma de sélection de cas de tests basé sur l'arbre de classification. Le but final étant de fournir des cas de test des IDS qui soient à la fois pertinents et représentatifs des différentes attaques.

4. Nouvelle classification

4.1. Objectifs

La définition d'une taxonomie systématique devrait passer, en réalité, par une définition judicieuse des principaux objectifs à respecter.

Tout d'abord, la classification, au même titre que la sélection des cas de test, doit être réfléchie et bien structurée. En effet, dans une sélection plus ou moins "aléatoire" des cas de test, les évaluateurs testent souvent leurs systèmes de manière ad hoc en utilisant quelques scripts disponibles sur Internet ou dans des listes de diffusions comme bugtraq [25]. Néanmoins, les scripts récupérés ne couvrent pas certains types d'attaques critiques et ne reflètent pas une distribution cohérente des attaques.

En outre, l'expression des résultats de l'évaluation en terme de ***classes*** d'attaques peut contribuer certainement à une meilleure compréhension de l'évaluation ainsi qu'à une représentation et interprétation plus précises de ses résultats. En effet, il est plus intéressant de dire que l'IDS est faible (ou robuste) vis-à-vis de la détection de tel ou tel type d'attaques. Dans le cas contraire, où on exprime les résultats en distinguant chaque attaque prise individuellement (et non de manière générique à travers les classes d'attaques), les conclusions peuvent être interprétées de manière biaisée.

Ceci étant, les classes résultantes ainsi que le processus de classification doivent respecter, le plus possible, les propriétés suivantes :

1. *complétude (*c-à-d *exhaustivité)* : un schéma de catégorisation doit tenir compte de toutes les attaques possibles (connues et inconnues) ;
2. *extensibilité* : quand de nouvelles attaques apparaissent, le schéma de catégorisation doit permettre de les classifier.
3. *clarté des critères* : le schéma et les règles de classification doivent être bien établies de manière à ce qu'une attaque puisse être classifiée en prenant une et une seule classe à partir de chaque dimension ;
4. *répétitivité* : la ré-application du processus de classification doit toujours produire les mêmes résultats, autrement dit, si on répète les étapes suivies pour la classification d'une certaine attaque, on doit toujours la placer dans la même catégorie ;
5. *conformité avec les standards* et terminologies existants, notamment avec les bases de données et dictionnaires des vulnérabilités [4, 15], qui sont actuellement largement utilisés ;
6. *exclusion mutuelle* : être sûr qu'une attaque ne fait pas partie de deux catégories différentes, une dimension n'aura donc que des classes mutuellement exclusives ;

Dans le cas de notre étude, il faudrait, en plus, garder une vision "évaluateur" (et non "attaquant") tout au long du processus de classification. Ceci va considérablement influencer la procédure de sélection de cas de tests nécessaires**.**

4.2. Notre classification

Commençons par analyser les attributs mentionnés dans la section 3, afin de n'en retenir que les plus pertinents d'un point de vue "évaluateur" ; ceux qui sont invisibles par l'IDS ou dénués de sens seront donc écartés.

Par exemple, des dimensions comme l'objectif de l'attaquant, ne seront pas traitées dans notre classification, d'autant plus qu'il est à la fois difficile et inutile d'imaginer l'intention de l'attaquant. Dans notre vision, toute tentative d'attaque est considérée comme une menace sérieuse, quelque soit l'objectif qui la sous-tend.

Dans le même sens, des dimensions comme le résultat de l'attaque ou la propriété de sécurité sont également moins pertinents dans notre étude. En effet, une fois que l'attaquant prend la main dans un système (en particulier s'il obtient l'accès root), il peut généralement modifier, détruire ou divulguer les informations, et donc porter atteinte à la fois aux propriétés de confidentialité, d'intégrité et de disponibilité.

Par ailleurs, nous estimons que les dimensions "type" et "technique de détection" ne servent pas à définir une catégorisation claire.

Comme indiqué dans la Figure 1, notre classification repose sur cinq dimensions. Ces dimensions sont sélectionnées de manière à couvrir les sources, les cibles et les manifestations des attaques, informations nécessaires et suffisantes pour le test des IDS. Ces dimensions sont les suivants :

- ***Source*** : indique l'endroit d'où l'attaque a été lancée. Elle possède deux classes : locale et distante.
- ***Privilège obtenu*** : nous distinguons quatre classes, les classes "root" et "utilisateur" signifient respectivement que l'attaquant a réussi à obtenir l'accès "root/administrateur", resp. "utilisateur" ; la classe "système" qui permet l'exécution de processus avec les privilèges "système" ; la quatrième classe "aucun" couvre les attaques qui n'ont besoin d'aucun privilège d'accès au système, ex. attaques de reconnaissance (*scans*).
- ***Vulnérabilité*** : d'un point de vue de l'évaluateur, il est intéressant d'exprimer la relation entre les attaques et les vulnérabilités exploitées ; ceci va en particulier aider à choisir (lors de la phase de test) les attaques qui exploitent ces vulnérabilités, et qui sont d'ailleurs répertoriées et disponibles dans des bases de données standardisées de vulnérabilités.
- ***Moyen*** par lequel l'attaque est lancée : trafic réseau, action exécutée directement sur la machine et qui n'apparaît pas dans l'interface réseau.
- ***Cible*** : qui peut être la mémoire, le système d'exploitation, la pile réseau, le système de fichier ou un processus.

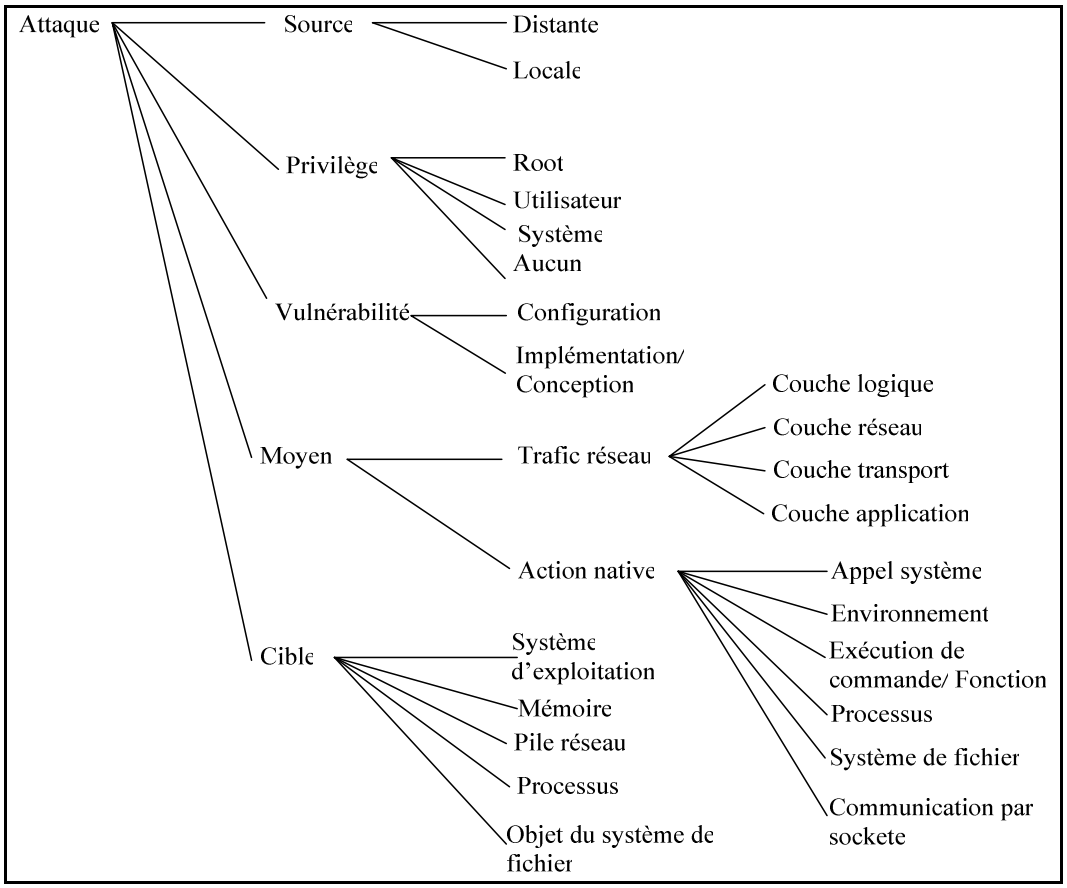

Figure 1 – Nouvelle classification : classes et attributs.

Remarquons que contrairement aux classifications existantes, notre taxonomie tient compte, non seulement des caractéristiques observables de l'attaque (comme c'est le cas des classifications orientée "IDS" [1] ou orientée "défense" [8]), mais aussi des aspects opérationnels, qui sont importants pour l'évaluateur.

En effet, la classification que nous proposons fournit les informations essentielles pour la génération des attaques et l'analyse des cas de test. Par exemple, la dimension "source" donne une idée sur l'endroit d'où l'attaque doit être générée pour le test ; de même, la dimension "vulnérabilité" donne une information sur la configuration à avoir (ou l'inverse) pour le test. Dans le même sens, la sévérité des attaques est implicitement décrite à partir de la dimension "privilège".

Il est également important de noter que notre classification respecte les objectifs (règles de bonne pratique) déjà identifiés dans la Section 4.1.

Tout d'abord, les cinq attributs que nous proposons sont choisis de façon à avoir une caractérisation exhaustive, couvrant différentes facettes des attaques. Ainsi, n'importe quelle attaque peut être caractérisée, c-à-d classifiée [propriété 1: *complétude*].

Par ailleurs, l'extensibilité des dimensions *"cible"* et *"moyen"* permet de classifier les nouvelles attaques (ex. notamment celles qui utilisent de nouveaux moyens ou visent de nouvelles cibles) [propriété 2 : *extensibilité*].

De plus, les définitions des dimensions que nous proposons aident amplement à déterminer (facilement) la classe de chaque attribut qui caractérise l'attaque [propriété 3 : *clarté des critères*].

Cette clarté des critères aide également à placer une certaine attaque dans la même catégorie si on ré-applique le schéma de classification [propriété 4: *répétitivité*].

En outre, la dimension "vulnérabilité" peut établir un lien direct entre l'attaque et une ou plusieurs entrées dans les bases des données standardisées des vulnérabilités (ex. CVE ou OSVDB) [propriété 5 : *conformité aux standards*].

Enfin, puisque les attributs de notre classification sont mutuellement exclusifs, une attaque ne peut, à priori, faire partie de deux catégories différentes [propriété 6 : exclusion mutuelle] ; cette propriété sera davantage démontrée lors de la classification d'attaques réelles existantes.

5. Schéma pour la sélection des cas de test

5.1. Spécification et génération des cas de test

Les arbres de classification (CTM pour *Classification-Tree Method*) sont des outils de classification supervisés qui ont été développés par Grotchmann et Grimm dans le domaine de l'ingénierie des logiciels [5]. Comme son nom l'indique, cette méthode représente graphiquement les partitions du domaine d'entrée sous forme d'arbre.

Ils déterminent des règles de classification en deux temps. Dans une première étape, une partition de l'espace des prédicteurs est déterminée de façon à ce que la distribution de la variable à prédire diffère le plus possible d'une classe à l'autre de la partition. On commence par partitionner les données selon les modalités de l'attribut le plus discriminant, puis on répète l'opération localement sur chaque noeud ainsi obtenu jusqu'à la réalisation d'un critère d'arrêt. Dans un second temps, après que l'arbre ait été généré, on dérive les règles de classification en choisissant la valeur de la variable à prédire la plus pertinente.

Dans notre cas, le but est de pouvoir former des cas de test en combinant des classes appartenant à différentes dimensions.

Dans une première étape, le domaine des entrées du test est d'abord considéré selon divers aspects ; pour chaque aspect, des classifications complètes et disjointes sont formées. Les classes résultantes sont, à leur tour, divisées en sous-classes (figure 2).

Dans la deuxième étape, une grille est dressée au-dessous de l'arbre. Chaque colonne de la grille contient les feuilles de l'arbre de classification (figure 2).

Un cas de test correspond, en fait, à une sélection d'une seule classe fille de chaque attribut/dimension de niveau supérieur ; en d'autres termes, chaque ligne de la grille indique un cas de test distinct.
Néanmoins, tous les cas de test possibles théoriquement par cette méthode, ne sont pas forcément valides ou intéressants. La personne qui planifie le test doit donc identifier les cas valides et éliminer les autres, en se basant notamment sur les contraintes ainsi que d'autres informations concernant le système comme l'explique l'exemple de la Section 5.2.

L'arbre de classification, présente plusieurs avantages. Tout d'abord, l'identification de tous les cas possibles ainsi que la sélection des cas de test pertinents se fait de manière systématique, ce qui facilite sa gestion et aide à réduire ou éliminer certaines erreurs. De plus, sa représentation graphique améliore la visualisation et facilite la communication entre les personnes qui font la spécification, ceux qui s'occupent de développement et ceux qui gèrent les tests.

5.2. Identification des cas de test

Une fois tous les cas de test possibles générés, il faut identifier les cas de test pertinents. Pour cela, nous avons utilisé l'outil CTE, pour *Classification Tree Editor* [3]. Cet outil permet, entre autres, l'automatisation de la génération et l'identification des cas de test.

Ainsi, à partir de notre schéma de classification et en utilisant la CTM, l'outil CTE génère toutes les combinaisons possibles des sous-classes. Ces combinaisons représentent les cas de tests d'attaques possibles.

Afin de couvrir l'espace des attaques, le nombre de combinaisons possibles est de l'ordre de quelques milliers (plus précisément, 1920) au lieu des 9600 si on utilise la classification d'Alessandri [1].

L'outil CTE permet l'application des contraintes sur l'arbre de classification. Cela aide à réduire davantage, à regrouper ou à réordonner les cas de test afin de n'en retenir que les plus pertinents pour l'évaluation en cours.

D'ailleurs, CTE offre un formalisme simple et puissant pour l'expression des contraintes en combinant des règles contenant des sous règles entre parenthèses (sous formes de prédicats), des connecteurs interpropositionnels tels que et (*), ou (+), non (NOT), etc.

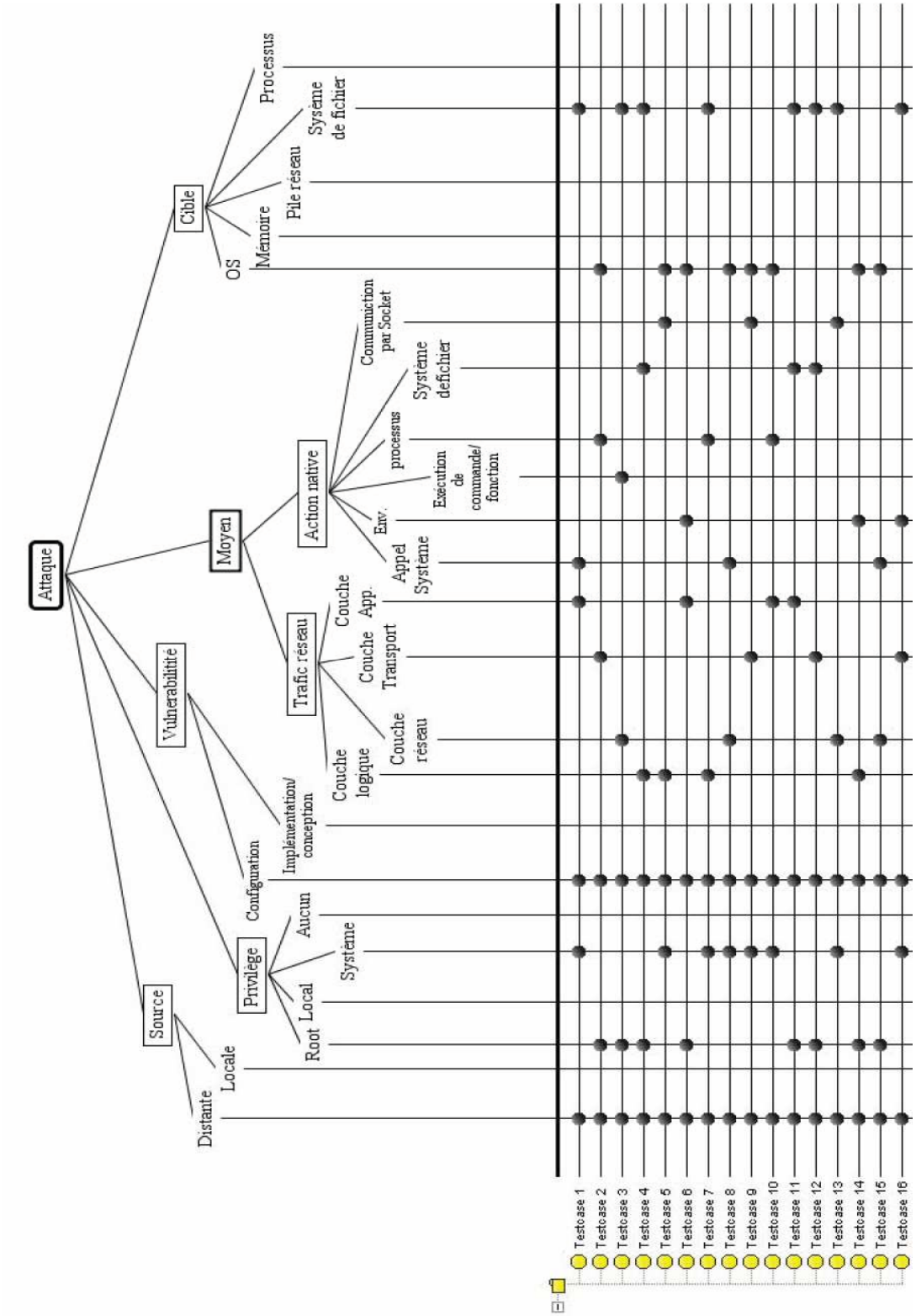

Figure 2 Exemple d'un arbre de classification ainsi que les cas de tests générés par CTE.

Par exemple, la contrainte suivante :

*Distante * (root + système) * Vul_configuration * Trafic réseau * (FS object + OS)*

génère des cas de test pour des attaques distantes qui exploitent des vulnérabilités introduites lors de la configuration, qui fournissent des accès "root" ou "système", qui sont visibles sur le trafic réseau, et qui visent le système d'exploitation ou le système de fichiers.

Comme indiqué dans la Figure 2, l'application de cette contrainte réduit les cas de test à 16. Par exemple, le premier cas de test concerne les attaques qui :

- sont lancées à distance,
- exploitent des vulnérabilités introduites lors de la configuration
- fournissent des accès "système",
- sont visibles sur le trafic réseau au niveau applicatif,
- utilisent des appels systèmes, et
- visent le système de fichiers.

Un autre exemple de cas de test qui représentent les attaques par déni de service (comme les « SYN flooding ») est donnée par la règle suivante :

*Distante * Aucun *Vul_ Implemenation/conception * Trafic réseau_transport * Pile réseau*

6. Conclusions et perspectives

Maintenant que les IDS sont des composants essentiels dans l'architecture des systèmes sécurisés, il devient de plus en plus nécessaire de pouvoir évaluer, avec certitude, leur robustesse vis-à-vis de tel ou tel type d'attaques. Ceci est extrêmement important dans la mesure où il va aider les administrateurs à :

- savoir si le niveau de sécurité de leur système est satisfaisant,
- identifier les points les plus critiques à surveiller ou à corriger dans les IDS étudiés,
- estimer s'il est rentable de mettre en œuvre telle ou telle défense supplémentaire,
- pouvoir comparer plusieurs IDS et déduire lequel est le plus adapté à leur système.

Néanmoins, l'évaluation des IDS souffre de certains obstacles, notamment, la multitude, la complexité, et la disparité des attaques (connues et non connues) ainsi que le manque de données réalistes précises qui peuvent être utilisées pour le test.

Dans cet article, nous avons d'abord mis en évidence la nécessité d'avoir une classification des attaques ainsi qu'une technique pour la génération et la sélection des cas de test.

Nous avons démontré que les classifications existantes ne sont pas tout à fait adaptées à l'évaluation et au test des IDS. Ceci est plus ou moins justifié car elles ont été conçues pour d'autres fins. Nous avons donc proposé une classification adaptée à l'évaluation des IDS ; celle-ci catégorise les attaques selon leurs caractéristiques observables par l'IDS, mais aussi selon les aspects opérationnels, points importants pour l'administrateur.

Ensuite, nous avons proposé une nouvelle approche pour sélectionner les cas de test les plus pertinents pour le système étudié. Cette approche tient compte de la classification déjà identifiée et des contraintes du système ; et utilise une technique basée sur l'arbre de classification (la méthode CTM). Dans notre proposition, la sélection des attaques ne se fait pas de manière aléatoire (comme c'est le cas de certaines évaluations qui utilisent des attaques collectées sur Internet ou à partir de listes de diffusions), mais de manière systématique qui tient compte de tout l'espace des attaques. De plus, l'adaptation de cette méthode à chaque type de systèmes (c-à-d en tenant compte des contraintes) réduit considérablement les cas de test pour ne retenir que les plus pertinents pour l'étude en cours.

Enfin, nous avons proposé d'utiliser l'outil CTE pour la génération et la sélection semi automatique des cas de test.

Dans la suite de ce travail, il conviendrait d'appliquer cette approche aux attaques et exploits connus (ex. bugtraq [25] et metasploit [26]) afin de les caractériser, les catégoriser et les répertorier. Ensuite, nous comptons étendre les fonctionnalités de CTE en ajoutant les attaques répertoriées, et nous en servir pour la sélection automatique des attaques pendant la phase de test de l'IDS. En effet,

CTE est actuellement générique et se limite à la sélection des cas de test sans tenir compte des instances d'attaques existantes.

7. Références

[1] Dominique Alessandri, "Attack-Class-Based Analysis of Intrusion Detection Systems", *Ph.D. Thesis*, Newcastle upon Tyne, UK: University of Newcastle upon Tyne, School of Computing Science, 2004.

[2] Matt Bishop, "Vulnerabilities Analysis", *Int. Symp. on Recent Advances in Intrusion Detection*, 1999.

[3] Classification Tree Editor "CTE", disponible à <http://systematic-testing.com>

[4] Common Vulnerabilities an Exposures "CVE", disponible à <http://cve.mitre.org/>

[5] M. Grochtmann, J. Wegener, K. Grimm, "Test case design using classification trees and the CTE", *In: Proceedings of the 8th International Software Quality Week*, pp. 1-11, 1995.

[6] John D. Howard. "An Analysis of Security Incidents on The Internet", *PhD thesis*, Carnegie Mellon University, 1997.

[7] K. Kendall, "A database of computer attacks for the evaluation of intrusion detection systems", *Master's Thesis*. Massachusetts Institute of Technology, Cambridge, MA, June 1999.

[8] K. S Killourhy; R. A. Maxion, and K. M. C. Tan, "A Defense-Centric Taxonomy Based on Attack" Manifestations, *In International Conference on Dependable Systems & Networks*, pp. 102-111,, Italy, 28 June - 01 July 2004.

[9] Sandeep Kumar, "Classification and Detection of Computer Intrusions", *PhD thesis*, Purdue, 1995.

[10] Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, and Kumar Das, "Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation", *Third Intl. Workshop on Recent Advances in Intrusion Detection (RAID2000)*, Toulouse, LNCS, vol. 1907, pp. 162-82.

[11] Ulf Lindqvist and Erland Jonsson, "How to Systematically Classify Computer Security Intrusions", *IEEE Security and Privacy*, pages 154–163, 1997.

[12] Daniel Lowry Lough, "A Taxonomy of Computer Attacks with Applications to Wireless Networks", *PhD thesis*, Virginia Polytechnic Institute and State University, 2001.

[13] John McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory", *ACM Transactions on Information and System Security*, Vol. 3, No. 4, pp. 262-294, November 2000.

[14] P. Mell, V. Hu, R. Lippmann, J. Haines, M. Zissman, "An Overview of Issues in Testing Intrusion Detection Systems", *NISTIR 7007*, National Institute of Standards and Technology, august 2003.

[15] Open Source Vulnerability Data Base: OSVDB: http:// www.osvdb.org/

[16] N. J. Puketza, K. Zhang, M. Chung, B. Mukherjee, and R. A. Olsson, "A Methodology for Testing Intrusion Detection Systems", *IEEE Trans. on Software Engineering*, vol. 22, pp. 719--29, October 1996.

[17] M. S. Gad El Rab, A. Abou El Kalam, "Testing Intrusion Detection Systems: An Engineered Approach", *IASTED International Conference on Software Engineering and Applications (SEA 2006), Nov. 2006.*

[18] Simon Hansmann, "A Taxonomy of Network and Computer Attacks", *Diplom Thesis*, University of Canterbury, Christchurch, New Zealand, Nov. 2003.

[19] D.J. Webar, "A Taxonomy of Computer Intrusions", *Master Thesis*, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1998.

[20] X-FORCE vulnerability data base, disponible à <http://xforce.iss.net/xforce/search.php>

[21] National Vulnerability Data Base, disponible à <http://nvd.nist.gov/>

[22] R. Lippmann, *et al.*, "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation", *DISCEX'00 - DARPA Information Survivability Conference & Exposition*, vol. 2, pp. 12—26, 2000.

[23] Peter G. Neumann and Donn B. Parker, "A Summary of Computer Misuse Techniques", *12th National Computer Security Conference*, Baltimore, MD, 1989, pp. 396—407.

[24] Snort *Intrusion Detection System*, disponible à <http://www.snort.org/>

[25] Bugtraq mail liste, disponible à : <http://www.securityfocus.com>

[26] Metasploit framework, disponible à : <http://www.metasploit.com/>