CVSS attack graphs

Laurent Gallon
LIUPPA
Université de Pau et des Pays de l'Adour
Mont de Marsan, France
Email : laurent.gallon@univ-pau.fr

Jean Jacques Bascou
LIUPPA
Université de Pau et des Pays de l'Adour
Mont de Marsan, France
Email : bascou@univ-pau.fr

***Résumé*—Dans cet article, nous proposons de modifier la définition des "attack models" et des "attack graphs", afin de rajouter des informations quantitatives à ces graphes, et plus particulièrement des notions de dommages, et de probabilité d'exploitation, à la fois pour chaque hôte du réseau cible, mais aussi pour ce réseau dans sa globalité. Les métriques utilisées pour calculer ces valeurs sont issus du framework CVSS et de la base de vulnérabilités NVD. Nous montrons ensuite, au travers d'un exemple, l'apport de notre contribution.**

I. Introduction

L'une des tâches importantes d'un administrateur est d'essayer de rendre son réseau le plus "sûr" possible, vis à vis des attaques qu'il peut subir. Pour cela, sa première tâche consiste à déterminer quelles sont les vulnérabilités potentiellement exploitables par un attaquant sur les hôtes de son réseau. Il existe aujourd'hui plusieurs bases de vulnérabilités, qui listent l'ensemble des vulnérabilités *connues*, c'est-à-dire qui ont été validées et confirmées par les organisations spécialisées et les éditeurs d'antivirus. On peut, par exemple, citer la National Vulnerability Database, qui est maintenue par le NIST [3]. A partir de cette base, et en utilisant un scanner de vulnérabilité (comme *nessus* par exemple [4]), l'administrateur peut obtenir la liste des vulnérabilités connues de son réseau, en fonction des matériels et des logiciels qui le constituent.

Mais cela n'est pas suffisant. En effet, la plupart du temps, les attaquants ne cherchent pas à exploiter une seule vulnérabilité sur un réseau cible, mais une succession de vulnérabilités, qui va leur permettre de progresser et d'atteindre un objectif bien précis. Il faut donc que l'administrateur soit aussi capable de rechercher les attaques "complexes" dans son réseau, constituées de l'exploitation d'une succession de vulnérabilités atomiques.

Les "attack models" et "attack graphs" ont été proposés dans ce but par Sheyner *et al.* [10][2][11]. Ils permettent de construire toutes les successions d'exploitation de vulnérabilités permettant à l'attaquant d'atteindre un objectif précis, qu'il ne pourrait pas atteindre en exploitant une vulnérabilité seule. Le principal défaut de ce modèle est qu'il ne donne aucune information quantitative sur les attaques potentielles (probabilités, dégâts, risque). Plusieurs travaux ont tenté de pallier à cela, comme par exemple dans [12][16], en essayant de définir des probabilités d'occurrence pour chaque attaque. Cependant, aucun article n'a véritablement traité le problème des dégâts causés aux hôtes et au réseau, ni la difficulté de mise en oeuvre de ces attaques. C'est justement ces aspects que nous abordons dans cet article. Nous nous basons sur le framework CVSS [14] pour rajouter aux attack graphs classiques une évaluation quantitative des dégâts et niveau d'exploitation des séquences d'attaques. Nous appelons le nouveau modèle CVSS attack graphs.

La suite de cet article est structurée comme suit : dans la section II, nous donnons un aperçu du framework CVSS et de l'évaluation quantitative des vulnérabilités. Dans la section III, nous détaillons la définition des attack models et attack graphs classiques. Puis, dans la section IV, nous définissons les CVSS attacks graphs. La section V est consacrée à un exemple mettant en exergue les apports de nos travaux. Enfin, la section VI conclue cet article, et donne des perspectives.

II. Introduction au framework CVSS

CVSS (Common Vulnerability Scoring System) [14][15][5] est un système ouvert permettant d'évaluer le niveau de gravité des vulnérabilités réseaux. Son objectif principal est d'associer un score de dangerosité, appelé score CVSS, à chaque vulnérabilité réseau. Pour cela, il s'appuie sur l'évaluation quantitative de différentes métriques. Ces métriques peuvent être regroupées en trois vecteurs (figure 1) :

- **vecteur de Base** : ces métriques représentent les caractéristiques intrinsèques de la vulnérablité.
- **vecteur temporel** : ces métriques réprésentent les caractéristiques de la vulnérabilité qui évoluent dans le temps, mais pas dans l'environnemment utilisateur.
- **vecteur environnemental** : représentent la prise en compte de l'environnement dans lequel la vulnérabilité est détectée.

Chaque partie, ou vecteur, donne lieu au calcul d'un score CVSS, comme montré sur la figure 2. Dans le calcul de ce score, seul le vecteur de base est obligatoire, les autres vecteurs sont optionnels. Dans la suite de cet article, nous n'utiliserons que la partie "Base" obligatoire de CVSS. En effet, les dictionnaires de vulnérabilités (CVE par exemple) ne renseignent que cette partie du framework CVSS. Seuls quelques constructeurs (comme CISCO par exemple) ajoute, uniquement pour certaines vulnérabilités (une minorité), la

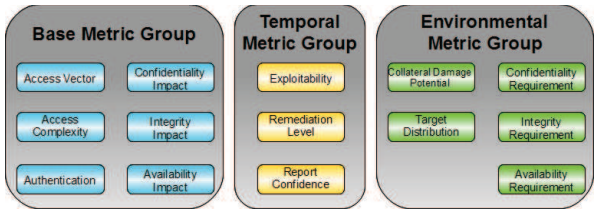

FIGURE 1. Répartition des métriques dans les différents vecteurs du framework CVSS

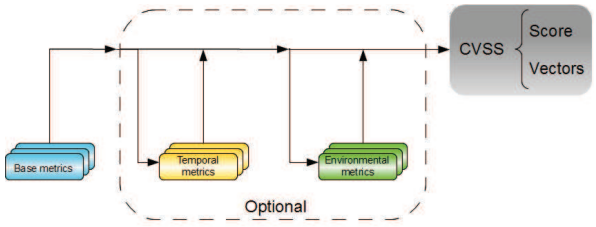

FIGURE 2. Calcul des différents scores CVSS

partie temporelle. Les données numériques que nous avons pu récupérer sont donc malheureusement limitées à la partie "Base".

A. Vecteur et métriques de base

Le vecteur de Base est constitué de six métriques de base. On peut les regrouper en deux sous-groupes :

- les métriques évaluant le niveau d'exploitation nécessaire au hacker pour exploiter la vulnérabilité :
  - *Access Vector AV* : le type d'accès réseau nécessaire pour exploiter la vulnérabilité
  - *Access Complexity AC* : la complexité de l'exploitation de la vulnérabilité
  - *Authentication AU* : le niveau d'authentification nécessaire pour exploiter la vulnérabilité
- les métriques mesurant l'impact de l'exploitation de la vulnérabilitié sur la sécurité de la cible :
  - *confidentiality impact CI*
  - *integrity impact II*
  - *availability impact AI*

Les valeurs qui peuvent être affectées à chacune de ces métriques sont données dans le tableau I.

A partir de ces différentes métriques, trois scores peuvent être calculés :

- le score d'exploitabilité (*Exploitability Score ES*), noté sur 10, qui permet d'évaluer la facilité d'exploitation de la vulnérabilité. Plus le score est grand, plus la vulnérabilité est considérée comme facilement exploitable. La formule permettant de calculer ce score est donnée ci-dessous :

$$Exploitability = 20 \times AV \times AC \times AU \quad (1)$$

- le score d'impact (*Impact Score IS*), noté sur 10, qui évalue l'impact de la vulnérabilité sur la cible de l'attaque :

$$Impact = 10.41 \times (1-(1-CI) \times (1-II) \times (1-AI)) \quad (2)$$

- le score de base (*Base Score BS*), noté sur 10, qui donne le niveau de dangerosité de l'attaque, et qui est calculé à partir du score d'exploitabilité et le score d'impact :

$$BaseScore = round\_to\_1\_decimal(((0.6 \times Impact) + (0.4 \times Exploitability) - 1.5) \times f(Impact)) \quad (3)$$

avec $f(impact) = 0$ si $Impact = 0$, 1.176 sinon.

AV	0.395 (Local)	0.646 (Adj. Net.)	1 (Network)
AC	0.35 (High)	0.61 (Medium)	0.71 (Low)
AU	0 (Multiple)	0.56 (Single)	0.704 (None)
CI/II/AI	0 (None)	0.275 (Partial)	0.66 (Complete)

TABLE I
VALEURS POSSIBLES DES MÉTRIQUES DE BASE

Métriques de base					
AV	AC	AU	CI	II	AI
Local Adj.Net. **Network**	High **Medium** Low	Multiple Single **None**	**Complete** Partial None	**Complete** Partial None	**Complete** Partial None
Base Vector → AV :N/AC :M/Au :N/C :C/I :C/A :C					
Score CVSS de base					
Exploitability score = 8.6			Impact score = 10.0		
Base score = 9.3					

TABLE II
EXEMPLE DE LA VULNÉRABILITÉ CVE-2006-6731

Le tableau II donne un exemple illustratif de l'utilisation de ces métriques et formules dans la base de vulnérabilités NVD [3]. La vulnérabilité CVE-2006-6731 qui y est illustrée, concerne des "buffer overflows" multiples dans l'environnement de développement Java JKD de Sun. Le tableau donne les valeurs choisies par les experts de NVD pour chaque métrique de base (les valeurs retenues sont celles en gras). On notera que l'exploitabilité est élevée (8.6/10), l'impact sur la cible est maximum (10/10), ce qui donne un score de base très élevé (9.3/10), et donc une vulnérabilité considérée comme très dangereuse.

Une des critiques principales que l'on peut faire au framework CVSS est qu'il évalue les vulnérabilités de manière individuelle, sans tenir compte du contexte, et notamment de l'exploitation antérieure d'autres vulnérabilités. En effet, dans la très grande majorité des cas, les attaques sur les systèmes d'information ne sont pas constituées de l'exploitation d'une seule vulnérabilité, mais au contraire d'une succession de plusieurs attaques, qui font progresser l'attaquant vers son objectif final. Dans CVSS, l'évaluation des métriques est fait sans tenir compte des dégâts et des privilèges déjà obtenus par l'attaquant. Nous proposons dans cet article une méthode pour rajouter les métriques CVSS dans le modèle "Attack Graph", afin d'obtenir une évaluation non pas d'une vulnérabilité seule, mais de la totalité de l'attaque.

III. "ATTACK MODELS" ET "ATTACKS GRAPHS"

Un "attack model" [1][9][10][11][12] est un outil qui permet de modéliser la structure d'un réseau (machines et connectivité), et les vulnérabilités des machines le constituant. Le modèle d'une vulnérabilité est constitué de préconditions d'exploitation (services actifs sur la machine cible, connectivité, privilèges) et de postconditions d'exploitation (nouveaux privilèges obtenus, impact sur les services et la connectivité de la machine cible).
A partir de l'"attack model", on peut construire des "attack graphs". Un "attack graph" est un graphe montrant toutes les combinaisons possibles de vulnérabilités (appelées *exécutions*) permettant à l'attaquant d'atteindre un objectif bien précis sur le réseau modélisé. Nous donnons ci-après une définition plus formelle, tirée des définitions données par Sheyner et Wing dans [2].

Définition 1 : *Un "attack model"* AM *est un automate fini* $AM = (S, \tau, s_0)$*, où* S *est un ensemble d'états,* $\tau \subseteq S \times S$ *est une relation de transition, et* $s_0 \in S$ *est l'état initial.* S *représente l'ensemble de trois agents* $I = \{E, D, N\}$*, où* E *est l'attaquant,* D *est l'ensemble des mécanismes de défense, et* N *est le système attaqué. Chaque agent* $i \in I$ *peut avoir différents états possibles.*

Dans cet article, nous nous focaliserons sur les "attack models" réseaux. D représente à la fois l'administrateur du réseau, et les outils de sécurité mis en oeuvre dans ce réseau, comme par exemple les IDS. N est composé de trois composants : H, qui est l'ensemble des hôtes du réseau, C qui donne la topologie du réseau, R la connectivité entre les différents hôtes (en particulier toutes les connexions autorisées ou refusées par le parefeu), et T qui exprime les relations de confiance entre les hôtes (par exemple, des relations d'approbations entre serveurs).

Définition 2 : *Une exécution finie* α *d'un "attack model" est une séquence finie d'états* $\alpha = s_0 s_1 ... s_n$*, telle que pour tout* $0 \leq i < n$*,* $(s_i, s_{i+1}) \in S$

Définition 3 : *soit une propriété de sécurité* P*, une exécution* α *est correcte vis à vis de* P *si* P *est vérifiée dans tous les états* $s_i \in \alpha$*. Une exécution* α *est incorrecte vis à vis de* P *si au moins un état* $s_i \in \alpha$ *ne respecte pas* P*.*

Définition 4 : *soit un "attack model"* AM *et une propriété de sécurité* P*, un "attack graph"* AG *de* AM *par rapport à* P *est l'ensemble des exécutions incorrectes de* AM *vis à vis de* P*.*

Un des inconvénients majeurs des "attack models" et des "attack graphs" est qu'ils ne donnent aucune information quantitative sur la dangerosité des différentes exécutions, ni sur les dégâts induits sur le réseau et ses hôtes. Dans ce papier, nous définissons la métrique hd_i (*Host Damages*) qui évalue les dégâts subis par chaque hôte H_i du réseau au cours d'une l'exécution. De manière similaire, nous définissons la métrique cep_i (*Cumulative Exploitability Probability*) pour évaluer le niveau d'exploitation nécessaire pour pouvoir utiliser l'hôte H_i dans une exécution. Enfin, nous définissons deux autres métriques globales au réseau, nd (*Network Damages*) et nep (*Network Exploitability Probability*), qui évaluent les dégâts subis par le réseau, et la probabilité d'exploitation de chaque exécution. L'évaluation de ces métriques est effectuée à partir des métriques CVSS de chaque vulnérabilité constituant l'exécution. Ces valeurs sont rajoutées dans la définition des "attack graphs", pour obtenir les "CVSS attack graphs".

IV. "CVSS ATTACK GRAPHS"

Dans ce papier, nous nous basons sur les définitions des "attack models" et "attack graphs" données par Sheyner dans [11].

A. "CVSS attack model"

Un "CVSS attack model" est très similaire à un "attack model" classique. Seule la définition de l'hôte H_i est modifiée, afin de rajouter les métriques CVSS associées aux vulnérabilités de chaque hôte (BV_{ij}, bs_{ij}), et de rajouter les vecteurs de métriques de mesure des dégâts et de niveau d'exploitation (dmg_i, exp_i) pour chaque hôte.

Définition 5 : Un hôte $H_i \in H$ est un tuple $(id_i, svcs_i, sw_i, vuln_i, dmg_i, exp_i)$ où :
- id_i est l'identifiant de l'hôte
- $svcs_i$ est la liste des services actifs sur l'hôte
- sw_i est la liste des logiciels installés sur l'hôte
- $vuln_i = \{v_{ij}\}$ est la liste des vulnérabilités connues de l'hôte H_i. Chaque vulnérabilité v_{ij} est un tuple $(id_{ij}, BV_{ij}, bs_{ij})$ où :
  - id_{ij} est l'identifiant de la vulnérabilité (référence CVE)
  - $BV_{ij} = (AV_{ij}, AC_{ij}, AU_{ij}, CI_{ij}, II_{ij}, AI_{ij})$ est le vecteur de base CVSS associé à la vulnérabilité (ce vecteur est tiré de la base de vulnérabilités NVD [3])
  - bs_{ij} est le score de base de la vulnérabilité, calculé à partir des formules données dans le framework CVSS [14], et le vecteur de base
- $dmg_i = (CD_i, ID_i, AD_i, hd_i)$ évalue les dommages subis par H_i pendant l'exécution. CD_i (resp. ID_i et AD_i) donne les dégâts cumulatifs sur la propriété de confidentialité (resp. intégrité et disponibilité). hd_i donne les dégâts subis par l'hôte H_i dans sa totalité, c'est-à-dire en combinant les dégâts CD_i, ID_i et AD_i. Ces différentes métriques sont calculées comme suit :
  - $CD_i = max(max(CI_k), max(CD_l \times T(H_i, H_l)))$ où $\{CI_k\}$ est l'ensemble des valeurs de CI pour les k attaques atomiques précédemment exploitées dans l'exécution, $\{CD_l\}$ est l'ensemble des valeurs de CD des l hôtes du réseau, et T la matrice des relations de confiance entre hôtes (la définition de T peut être trouvée dans [2] par exemple).
  - $ID_i = max(max(II_k), max(ID_l \times T(H_i, H_l)))$

- $AD_i = max(max(AI_k), max(AD_l \times T(H_i, H_l)))$
- $hd_i = 10.41 \times (1-(1-CD_i) \times (1-ID_i) \times (1-AD_i))$

hd_i peut varier de 0 à 10. Tant qu'aucune vulnérabilité n'a été exploitée, $\forall i, hd_i = 0$. Le lecteur pourra trouver plus de détails sur la définition de dmg_i dans [7][8].

- $exp_i = (CAV_i, CAC_i, CAU_i, cpe_i)$ où
  - $CAV_i = min(AV_k)$: valeur la plus faible de la métrique AV pour les k vulnérabilités exploitées sur l'hôte H_i (plus AV est faible, plus le type d'accès à l'hôte est contraignant). Autrement dit, CAV donne le niveau d'accès nécessaire pour l'attaquant sur l'hôte H_i pour pouvoir exploiter les k vulnérabilités
  - $CAU_i = min(AU_k)$: valeur la plus faible de la métrique AU pour les k vulnérabilités exploitées sur l'hôte H_i (plus AU est faible, plus le niveau d'authentification exigé est grand). Autrement dit, CAU donne le niveau d'authentification nécessaire pour l'attaquant sur l'hôte H_i pour pouvoir exploiter les k vulnérabilités
  - $CAC_i = min(AC_k)$: valeur la plus faible de la métrique AC pour les k vulnérabilités exploitées sur l'hôte H_i (plus AC est faible, plus la complexité de l'exploitation de la vulnérabilité est grande). Autrement dit, AC donne le niveau d'expertise requis pour l'attaquant pour pouvoir exploiter les k vulnérabilités.
  - *Cumulative Exploitability Probability* $cep_i = 2 \times CAV \times CAC \times CAU$ évalue la difficulté d'exploitation des k vulnérabilités sur l'hôte H_i. Ce score peut varier de 0 à 1. Tant qu'aucune vulnérabilité n'a été exploitée sur l'hôte H_i, $cep_i = 1$.

B. "CVSS attack graphs"

A partir d'un "CVSS attack model", il est possible de construire des "CVSS attack graphs". Un "CVSS attack graph" est un "attack graph" classique, dans lequel nous utilisons les informations quantitatives issues du "CVSS attack model". Lors de cette construction, nous rajoutons le calcul de deux nouvelles métriques :

- *network damages nd* qui évalue les dommages subis par l'ensemble du réseau, et non pas un seul hôte. nd peut être calculé de différentes manières, en fonction de ce que les administrateurs du réseau considèrent comme critique :
  - dommages les plus grands subis par un hôte : $\forall i, nd = max(hd_i)$
  - valeur moyenne des dommages subis par les différents hôtes : $nd = mean(hd_i)$
  - dommages subis par un hôte important du réseau (un serveur par exemple) : $nd = hd_k$
  - n'importe quelle combinaison pondérée des dommages subis par les hôtes du réseau : $nd = sum(\alpha_i.hd_i)/sum(\alpha_i)$

  La définition de nd doit donc être choisie par les administrateurs, avant de construire les "CVSS attack graphs". Dans ce papier, nous avons choisie d'utiliser la définition la plus simple : $nd = mean(hd_i)$.
- *network exploitability probability nep* qui évalue le niveau d'expertise nécessaire à l'attaquant pour être capable d'exploiter l'exécution. Plus précisément, nep est basé sur le niveau de difficulté le plus grand auquel l'attaquant est confronté, en termes de Access Vector AV, Access Complexity AC et Authentication AU, non pas pour une seule vulnérabilité, mais pour la totalité des vulnérabilités constituant l'exécution. Nous définissons :
  - *Global Access Vector* : $\forall i, GAV = min(CAV_i)$
  - *Global Access Complexity* : $\forall i, GAC = min(CAC_i)$
  - *Global Authentication* : $\forall i, GAU = min(CAU_i)$

  A partir de ces 3 métriques, la valeur de nep est calculée comme suit :

$$nep = 2 \times GAV \times GAC \times GAU$$

L'utilisation de ces définitions nous permet donc de construire des "CVSS attack graphs", qui contiennent à la fois des informations qualitatives (privilèges gagnés par l'attaquant, services actifs où désactivés par l'attaque, connectivité entre machines) et des informations quantitatives (dommages et niveau d'exploitation à la fois pour chaque hôte, et pour le réseau dans sa globalité).

Dans sa définition historique, un "attack graph" donne toutes les combinaisons d'exploitation de vulnérabilités, permettant à l'attaquant d'atteindre un certain objectif. Cet objectif est traditionnellement une combinaison de privilèges obtenus par l'attaquant sur un ou plusieurs hôtes. Nous appelons ce type d'objectif un *objectif qualitatif*. Dans les "CVSS attack graphs", il est possible d'utiliser deux autres types d'objectifs :

- un *objectif quantitatif* : on ne se focalise que sur une combinaison des informations quantitatives contenues dans les "CVSS attack graphs". Par exemple, on peut vouloir obtenir toutes les combinaisons permettant d'obtenir des dommages sur le réseau supérieurs à un certain seuil : $nd \geq L$
- un *objectif quantitatif et qualitatif*, dans lequel on mélange des informations qualitatives et quantitatives. Par exemple, on peut vouloir obtenir toutes les exécutions permettant à l'attaquant d'obtenir les privilèges root sur un hôte H_3 et des dégâts supérieurs à un certain seuil sur ce même hôte : $root(H_3)$ *and* $hd_3 \geq L$

Ces deux derniers types d'objectifs ne pouvaient pas être utilisés dans les travaux antérieurs aux nôtres. Ils permettent d'étendre le pouvoir d'analyse et d'investigation des "attack graphs". Nous considérons que ce point est un des apports importants de nos travaux.

V. Exemple

Notre exemple est largement inspiré des exemples utilisés par Sheyner dans [10] et Ghosh dans [13]. Pour des raisons de longueur de l'article, il ne nous est pas possible de le détailler complètement. Le lecteur trouvera tous les détails dans [7][8]. Nous ne donnons ici que les éléments indispensables à une bonne compréhension de notre contribution.

La structure du réseau est donnée sur la figure 3. Il est composé de cinq hôtes : la machine attaquante (H_A), un

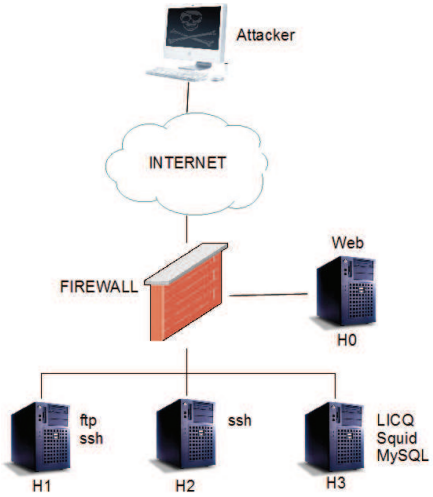

FIGURE 3. Exemple de réseau

Host	Name	Description	AV AC AU	AI CI II
H_0	iisbof	IIS buffer overflow	N L N	P P P
H_1	rhost	ftp rhost overwrite	N M N	P N N
	rsh	rsh login	N L S	P P P
H_2	nns	netbios null session	N L N	P N N
H_3	rtu	LICQ remote to user	N L N	P P P
	locbof	Local buffer overflow	L L N	C C C

TABLE III
LISTE DES VULNÉRABILITÉS CONNUES SUR LES HÔTES DU RÉSEAU

Vulnerability	Preconditions Intruder	Preconditions Network	Effects Intruder	Effects network
iisbof(S,T)	$user(S)$ $\neg root(T)$	$w3_T$ $R(S,T,80)$	$root(T)$	$\neg w3_T$
rsh(S,T)	$user(S)$ $\neg user(T)$	ftp_T $R(S,T,21)$	$Trust(T,S)$	
rhost(S,T)	$user(S)$ $\neg user(T)$	ftp_T $R(S,T,21)$	$user(T)$	

TABLE IV
MODÉLISATION DES VULNÉRABILITÉS

serveur Web (H_0), un serveur de domaine Windows (H_1), un poste client Windows (H_2), et un serveur Linux (H_3). La liste des vulnérabilités connues sur chaque hôte est donnée dans la table III. Les vecteur de base CVSS associés à chacune de ces vulnérabilités ont été trouvés dans la base de vulnérabilités NVD [3].

Chaque vulnérabilité est modélisée dans l'"attack model" correspondant au réseau. En particulier, on doit modéliser les préconditions d'exploitation des vulnérabilités, et leurs postconditions, c'est-à-dire leurs effets sur les hôtes et le réseau. Nous donnons en exemple les préconditions et postconditions de 3 vulnérabilités dans le tableau IV. Dans le cas de la vulnérabilité $rsh(S,T)$, la fonction $Trust$ indique que le principal effet de cette vulnérabilité est la mise en place d'une relation de confiance (approbation, ...) entre la machine cible et la machine source. On considère alors que l'attaquant bénéficie des mêmes privilèges sur la machine cible, que sur la machine source.

Sur la figure 3, le pare-feu limite la connectivité entre les hôtes du réseau. Par exemple, il autorise les machines internes (de H_1 à H_3) à accéder à la DMZ (machine H_0) seulement par le port 80. La fonction $R(S,T,p)$ dans le tableau IV est vraie si la connectivité entre la machine source S et la machine cible T est autorisée par le pare-feu pour le port p (la fonction R prend aussi en compte les pare-feus des hôtes s'ils en ont). De plus, les hôtes internes ne peuvent pas accéder au réseau externe directement (ils doivent passer par le serveur Web). Enfin, nous considérons qu'il n'y a pas, au départ, de relation d'approbation ou de confiance entre les hôtes du réseau.

Comme nous l'avons dit précédemment, nous pouvons construire une multitude de "CVSS attack graphs" à partir de la modélisation du réseau. Il n'est pas possible de présenter tous les cas dans cet article. Nous nous focalisons donc sur un exemple simple de "CVSS attack graphs" construit à partir d'un objectif qualitatif. L'objectif choisi est *"l'attaquant n'a pas obtenu les privilèges root sur* H_3*"* : $\neg root(H_3)$. Le graphe obtenu est de grande taille, aussi nous ne donnons, sur la figure 5, qu'une partie de ce dernier, plus exactement toutes les "branches" (exécutions) qui commencent par la séquence $iisbof(A,0)/rtu(0,3)$. Chaque rectangle en gras représente un état dans lequel l'attaquant a obtenu les privilèges *root* sur H_3.

Dans chaque état, on trouve plusieurs informations :
- dans la partie haute du rectangle, on trouve la liste des privilèges obtenus par l'attaquant sur les hôtes du réseau. $root(X)$ (respectivement $user(X)$) signifie que l'attaquant a obtenu les privilèges $root$ (respectivement $user$) sur l'hôte X.
- dans la partie médiane du rectangle, on trouve les changements dus à l'exploitation de la dernière vulnérabilité, par rapport au dernier état, en termes de *cumulative host damage hd / cumulative exploitability probability cep*
- dans la partie basse du rectangle, on trouve les changements induits par l'exploitation de la dernière vulnérabilité sur le réseau, en termes de *network damages nd / network exploitability property nep*

La figure 4 montre l'"attack graph" classique correspondant au même objectif, à des fins de comparaison.

Le tableau V détaille les 8 états du "CVSS attack graph" dans lequel l'attaquant a atteint son objectif. Certains de ces états peuvent être atteints par plusieurs exécutions (ce nombre est précisé dans la colonne "Exécution"). Pour chaque état, nous précisons :
- les privilèges gagnés par l'attaquant sur les différents hôtes du réseau
- les valeurs de hd / cep, qui donnent le niveau de dommages et d'exploitation pour chaque hôte H_i
- les valeurs de nd / nep qui donnent le niveau de dom-

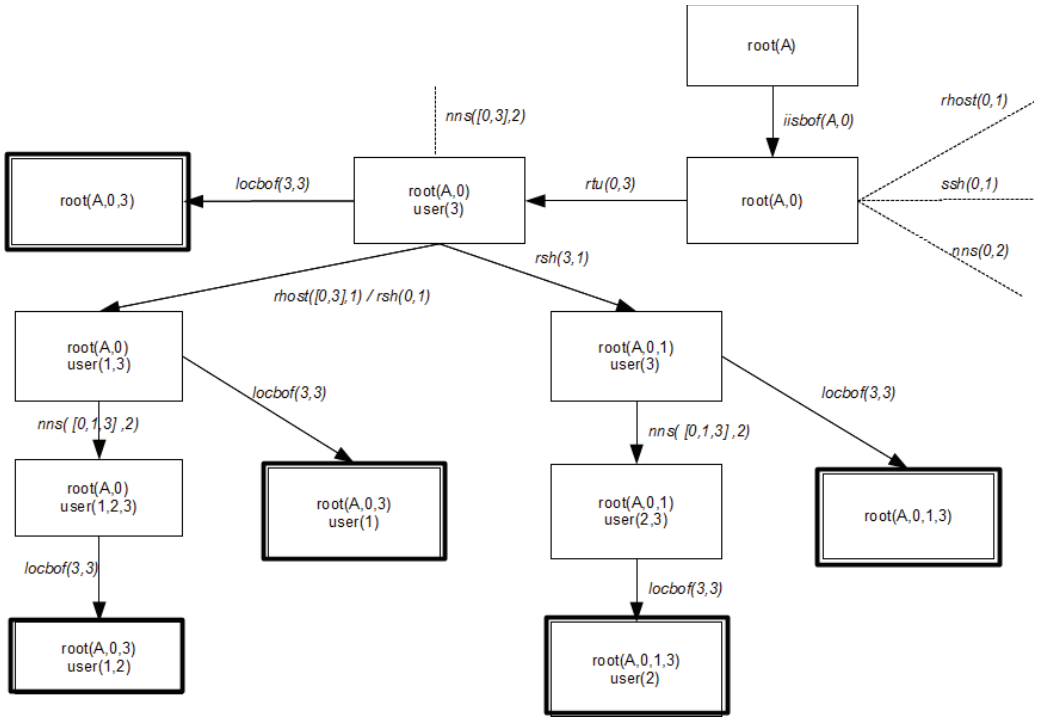

FIGURE 4. "Attack graph" classique, basé sur la propriété "l'attaquant a gagné les privilèges root sur l'hôte H_3"

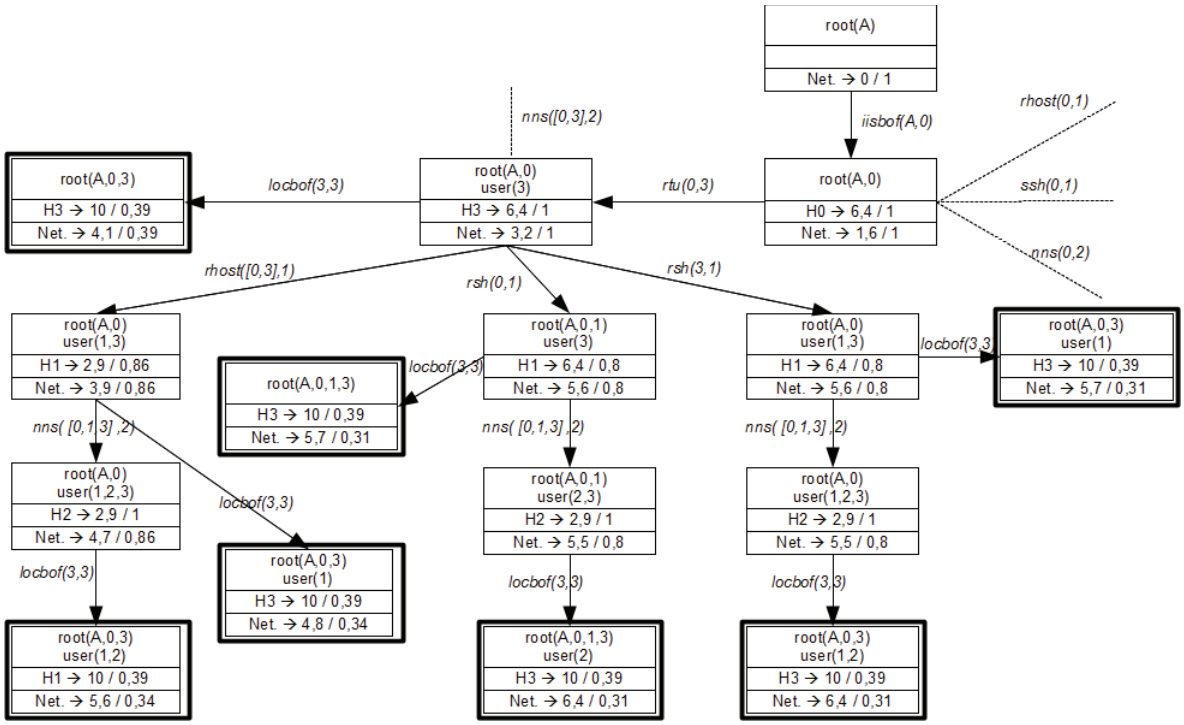

FIGURE 5. "CVSS attack graph", basé sur la propriété "l'attaquant a gagné les privilèges root sur l'hôte H_3"

Exécution	#	Privilèges	H_0	H_1	H_2	H_3	Réseau
$iisbof(A,0)/rtu(0,3)/locbof(3,3)$	1	R(A,0,3)	6.4/1	-	-	10/0.39	4.1/0.39
$iisbof(A,0)/rtu(0,3)/rhost([0,3],1)/locbof(3,3)$ (1 autre exécution amène à cet état)	2	R(A,0,3) U(1)	6.4/1	2.9/0.86	-	10/0.39	4.8/0.34
$iisbof(A,0)/rtu(0,3)/rhost([0,3],1)/nns([0,1,3],2)/$ $locbof(3,3)$ (4 autres exécutions amènent à cet état)	3	R(A,0,3) U(1,2)	6.4/1	2.9/0.86	2.9/1	10/0.39	5.6/0.34
$iisbof(A,0)/rtu(0,3)/rsh(3,1)/locbof(3,3)$	4	R(A,0,3) U(1)	6.4/1	6.4/0.8	-	10/0.39	5.7/0.31
$iisbof(A,0)/rtu(0,3)/rsh(3,1)/nns([0,1,3],2)/$ $locbof(3,3)$ (3 autres exécutions amènent à cet état)	5	R(A,0,3) U(1,2)	6.4/1	6.4/0.8	2.9/1	10/0.39	6.4/0.31
$iisbof(A,0)/rtu(0,3)/rsh(0,1)/locbof(3,3)$ (1 autre exécution amène à cet état)	6	R(A,1,0,3)	6.4/1	6.4/0.8	-	10/0.39	5.7/0.31
$iisbof(A,0)/rtu(0,3)/rsh(0,1)/nns([0,1,3],2)/$ $locbof(3,3)$ (5 autres exécutions amènent à cet état)	7	R(A,1,0,3) U(2)	6.4/1	6.4/0.8	2.9/1	10/0.39	6.4/0.31
$iisbof(A,0)/rtu(0,3)/nns([0,3],2)/locbof(3,3)$	8	R(A,0,3) U(2)	6.4/1	-	2.9/1	10/0.39	4.8/0.39

TABLE V
ETATS DES HÔTES ET DU RÉSEAU DANS LES ÉTATS OÙ L'ATTAQUANT A ATTEINT SON OBJECTIF

mages et d'exploitation pour le réseau

La première analyse montre que le niveau de dégâts nd subis par le réseau varie assez sensiblement d'une exécution à l'autre : de 4.1 pour l'état $\#1$, à 6.4 pour les états $\#5$ et $\#7$. La variation est de plus de 50%, ce qui prouve bien que les exécutions n'ont pas du tout le même impact sur le réseau. En effet, même si les objectifs qualitatifs sont les mêmes pour l'attaquant dans les deux exécutions, certaines vulnérabilités utilisées diffèrent, ce qui engendre un impact différent sur le système attaqué.

Si on regarde plus particulièrement les états $\#2$ et $\#4$, les privilèges obtenus par l'attaquant sont les mêmes ($root(A,0,3)$ $user(1)$), mais les mesures quantitatives effectuées donnent des résultats différents. Les dommages sont moins importants dans l'état $\#2$ ($nd = 4.8$) que dans l'état $\#4$ ($nd = 5.7$). En ce qui concerne le niveau d'exploitation nep, c'est l'inverse : dans l'état $\#2$, on a $nep = 0.34$, alors que dans l'état $\#4$, on a $nep = 0.31$. Ces différences proviennent de l'exploitation de la vulnérabilité $rhost$ dans l'exécution menant à l'état $\#2$, plutôt que l'exploitation de la vulnérabilité rsh dans l'exécution menant à l'état $\#4$. Ces deux vulnérabilités permettent à l'attaquant d'obtenir les mêmes droits $user$ sur la machine H_1, mais avec un impact et une exploitabilité différente (la vulnérabilité rsh est considérée, dans CVSS, comme plus dangereuse que la vulnérabilité $rhost$).

Dans les "attack graphs" classiques, ces deux états seraient regroupés en un seul (mêmes privilèges gagnés par l'attaquant). Ici nous voyons que l'exécution menant à l'état $\#4$ est plus "dangereuse" que les deux exécutions menant à l'état $\#2$, puisque le niveau d'exploitation nécessaire est plus faible, et que les dommages sont plus importants. Cet exemple illustre parfaitement l'apport des "CVSS attack graphs", c'est-à-dire la possibilité d'évaluer le niveau de dangerosité des différentes exécutions, pour savoir lesquelles sont les plus critiques. Bien entendu, ces analyses ne sont pas possibles sans l'utilisation des métriques CVSS.

Si on regarde le couple nd / nep, on se rend compte que l'état $\#1$ fait apparaître le niveau de dommages le plus faible ($nd = 4.1$), et le niveau d'exploitation le plus élevé ($nep = 0.39$), ce qui indique que c'est l'exécution la plus "difficile" à mettre en oeuvre. A contrario, les états $\#5$ et $\#7$ font apparaître le niveau de dégâts le plus grand ($nd = 6.4$) et le niveau d'exploitation le plus faible ($nep = 0.31$). On a donc affaire, dans le premier cas, à une exécution menée par un expert, qui "cible" véritablement son objectif, en essayant d'être le plus furtif possible (essaie de provoquer le moins de dégâts possible sur le réseau, pour diminuer au maximum le risque de détection, tout en atteignant son objectif), alors que dans les deux autres exécutions, l'attaque est faite pour induire le plus de dégâts possible. La figure 6 compare le niveau de dégâts et le niveau d'exploitation dans chaque état. La troisième courbe sur cette figure montre la variation de la métrique $\|nd - 10 \times nep\|$. Elle permet, **sur cet exemple précis**, de bien voir le niveau de furtivité offert par chaque exécution à l'attaquant : si cette valeur est faible (0.2 pour l'état $\#1$, 0.9 pour l'état $\#8$), alors l'attaque est relativement furtive. Si cette valeur est élevée (3.3 pour les états $\#5$ et $\#7$), l'attaque est beaucoup plus visible, et donc plus détectable. Nous pensons que le couple nd / nep (au travers, par exemple, de l'utilisation de la variable $\|nd - 10 \times nep\|$) est un bon indicateur de recherche de furtivité par l'attaquant, et donc du niveau d'expertise de ce dernier.

Enfin, comme nous l'avons montré sur le tableau V, le "CVSS attack graph" donne 8 états différents dans lequel l'attaquant à atteint son objectif. Ces 8 états font apparaître 6 combinaisons différentes de privilèges obtenus par l'attaquant, ce qui indique que l'"attack graph" classique qui est construit à partir de la même propriété (même objectif de l'attaquant) ne ferait apparaître que 6 états différents. Nous voyons donc apparaître ici le principal défaut de notre proposition : l'amplification de l'explosion combinatoire du nombre d'états, phénomène déjà pointé du doigt dans les "attack graphs" classiques. Nous reviendrons sur ce point dans les perspectives.

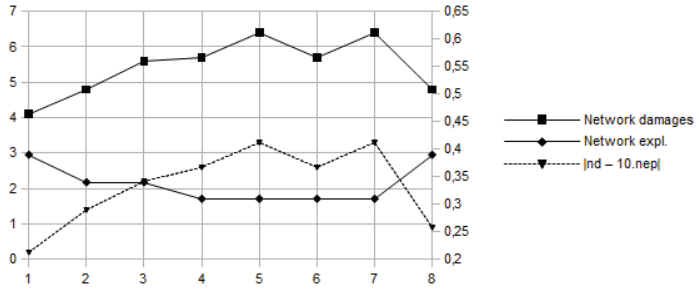

FIGURE 6. Comparaison du niveau de dégâts et du niveau d'exploitation

VI. CONCLUSION

Dans cet article, nous proposons une méthodologie pour rajouter les informations issues du framework CVSS dans les "attack graphs", et ainsi définir les "CVSS attack graphs". Ceci nous autorise à évaluer les dommages induits par les attaques sur les hôtes et le réseau, ainsi qu'à évaluer le niveau d'exploitation nécessaire pour pouvoir mettre en oeuvre ces attaques. Ces informations sont ensuite utilisées pour construire 3 grands types de "CVSS attack graphs", en se focalisant sur des objectifs qualitatifs, quantitatifs, ou un mélange de qualitatif et quantitatif. L'apport de ces informations est montré au travers d'un exemple basé sur un objectif qualitatif, qui permet de comparer les différences entre "attack graphs" et "CVSS attack graphs".

Dans un futur proche, plusieurs travaux vont être menés pour compléter ceux que nous avons présenté ici :

- nous avons vu que le problème initial de l'explosion combinatoire des "attack graphs" est amplifié par notre approche. Notre objectif est de ne pas générer plus d'états que les "attack graphs" classiques. Pour cela, nous pensons travailler sur la notion de classe d'états. Une classe d'états pourrait regrouper tous les états faisant apparaître les mêmes privilèges gagnés par l'attaquant. On aurait donc autant de classes d'états que d'états dans les "attack graphs" classiques. Les métriques quantitatives seraient regroupées en intervalles de valeurs, pour pouvoir prendre en compte toutes les possibilités dans une même classe d'états.
- nous souhaitons pouvoir définir une notion de "risque" à partir des métriques de dommages et d'exploitation. Dans une première étude, nous avons appliqué la définition "simpliste" $risk = damages \times exploitability$. Les résultats ne sont pas convaincants. Nous devons donc explorer plus en avant cette définition
- l'évaluation de la furtivité des attaques nous semble relativement intéressante, pour pouvoir ensuite confronter les "CVSS attack graphs" aux outils de détection d'attaques (IDS). Cependant, la définition que nous avons donnée est *ad hoc*, et ne peut certainement pas être appliquée dans le cas général. Une investigation est donc nécessaire pour proposer une meilleure définition

RÉFÉRENCES

[1] Lippmann, R. P., and Ingols, K. W., *An Annotated Review of Past Papers on Attack Graphs*, project report, Massachussets Institute of Technology (MIT), Lexington Lincoln Laboratory, March 2005.

[2] Sheyner, O., and Wing, J., *Tools for generating and analyzing attack graphs*, Second International Symposium on Formal Methods for Components and Objects, Leiden, The Netherlands, November 2003.

[3] National Vulnerability Database (http ://nvd.nist.gov/).

[4] Nessus : the Tenable Vulnerability Scanner tool (http ://www.tenable.com/products/nessus).

[5] Gallon, L., *On the impact of environmental metrics on CVSS scores*, The Second IEEE International Conference on Privacy, Security, Risk and Trust (PASSAT-2010) - Symposium on Secure Computing (SecureCom-10) Minneapolis, Minnesota, USA, August 2010.

[6] Gallon, L., *Vulnerability discrimination using CVSS framework*, 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS 2011) Paris, France, February 2011.

[7] Gallon, L. and Bascou, JJ., *Using CVSS in attack graphs*, The Sixth International Conference on Availability, Reliability and Security (ARES 2011) Vienna, Austria, August 2011.

[8] Gallon, L. and Bascou, JJ., *CVSS attack graphs*, The 7th International Conference on Signal Image Tehcnology & Internet Based Systems (SITIS 2011) Dijon, France. November 2011.

[9] Jha, S., Sheyner, O. and Wing, J., *Two formal analyses of attack graphs*, 15th IEEE Computer Security Foundations Workshop (CSFW'2002), Nova Scotia, Canada, June 2002.

[10] Sheyner, O., Haines, J., Jha, S., Lippmann, R. and Wing, J., *Automated Generation and Analysis of Attack Graphs*, IEEE Symposium on Security and Privacy (S&P'2002), Berkeley, California, USA, May 2002.

[11] Sheyner, *Scenario Graphs and Attack Graphs*, PhD thesis, Carnege Mellon University, 2004.

[12] Mehta, V., Bartzis, C., Zhu, H., Clarke, E. and Wing, J., *Ranking attack graphs*, 9th International Symposium on Recent Advances in Intrusion Detection (RAID'2006), Hamburg, Germany, September 2006.

[13] Ghosh, N. and Ghosh S.K. *An Intelligent Technique for Generating Minimal Attack Graph*, First Workshop on Intelligent Security (Security and Artificial Intelligence), SecArt '09, Thessaloniki, Greece, September, 2009

[14] Mell, P., Scarfone, K., and Romanosky, S., *A Complete Guide to the Common Vulnerability Scoring System (CVSS) Version 2.0*, Forum of Incident Response and Security Teams (http ://www.first.org/cvss/cvss-guide.html), June 2007.

[15] Scarfone, K. and Mell, P., *An analysis of CVSS version 2 vulnerability scoring*, 5th International Workshop on Security Measurement and Metrics (MetriSec'09), Orlando Florida, USA, October 2009.

[16] Wang, L., Islam T., Long. T., Singhal A. and Jajodia S., *An attack graph-based probabilistic security metric*, 22nd annual IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec 2008), London, UK, july 2008.