Rapport technique

Analyse commentée de l'article : "Adaptive, Model-based Monitoring for Cyber Attack Detection"

Alfonso VALDES, Keith SKINNER, RAID 2000, pp.80-92

Nathalie Dagorn[1], Yann Dagorn[1]

[1] Université de Nancy1
Laboratoire Lorrain de Recherche en Informatique et ses Applications (LORIA)
Campus Scientifique BP 239, F-54506 Vandœuvre-lès-Nancy Cedex, France
nathalie.dagorn@loria.fr
yann.dagorn@loria.fr
http://www.loria.fr/

Résumé. Les méthodes d'inférence pour la détection d'attaques sur des ressources d'information mettent en œuvre des techniques d'analyse de signature (détection d'abus) ou des méthodes statistiques (détection d'anomalie). L'analyse de signature a l'avantage de spécifier les attaques de manière précise, mais ne détecte que des attaques connues ; les méthodes statistiques sont capables de détecter les attaques de manière probabiliste, permettant l'extension aux attaques inconnues, mais les modèles sur lesquels est basée la détection sont parfois imprécis. Dans cet article, Valdes et Skinner présentent une technique performante et adaptative, utilisant les réseaux bayésiens pour analyser des rafales de trafic. Les classes d'attaques sont représentées comme des hypothèses de modélisation, continuellement renforcées par adaptation. Cette approche rassemble à la fois les meilleures fonctionnalités des techniques d'analyse de signature et celles des modèles statistiques : spécificité de la modélisation, adaptabilité et potentiel de généralisation. Le prototype de détection initial examine les en-têtes TCP et communique en IDIP (*Intrusion Detection Internet Protocol*). La technique d'inférence utilisée est également appropriée pour la corrélation de plusieurs détecteurs.

Mots-clés. Détection d'intrusion, approche innovante, coopération d'IDS, réseau bayésien.

Introduction

A ce jour, deux principales techniques d'inférence sont utilisées par les systèmes de détection d'intrusion (IDS). Dans l'*analyse de signature*, les descriptions d'attaques connues sont encodées sous forme de règles. Les systèmes statistiques « apprennent » un comportement normal à partir de données d'apprentissage, puis, en phase de

détection, génèrent des alertes pour rapporter les anomalies suspectées. Cet article décrit la structure et le fonctionnement d'eBayes TCP, qui applique comme technique d'inférence les méthodes bayésiennes.

eBayes TCP a été développé comme composant du système EMERALD[1]. L'innovation apportée par eBayes TCP est qu'il combine les meilleures fonctionnalités de la détection d'intrusion basée sur les signatures et celles de la détection d'anomalie : comme les mécanismes de signature, il peut intégrer des modèles d'attaques, mais a la capacité de s'adapter à l'évolution des systèmes. A l'instar des mécanismes probabilistes, il a le potentiel de détecter des attaques non encore connues. En outre, le système inclut une capacité d'adaptation, qui permet au système de générer des modèles cohérents à partir d'états non connus.

eBayes TCP analyse les sessions TCP, définies comme des rafales de trafic temporellement contiguës provenant d'un client IP donné. Il n'est pas très important pour le système de démarquer exactement les sessions. L'analyse est effectuée par inférence bayésienne à des intervalles périodiques dans une session, où l'intervalle est mesuré en nombre d'événements ou en temps écoulé (l'inférence est toujours mise en œuvre lorsque le système croit qu'une session s'est terminée). Entre les intervalles d'inférence, l'état du système est propagé selon un modèle de Markov. Après chaque inférence, le système écrit un rapport et génère des alertes IDIP (*Intrusion Detection Internet Protocol*) pour les sessions suspectes.

Inférence bayésienne

1) La connaissance (*knowledge*) est représentée sous forme de nœuds dans un arbre ; chaque nœud peut prendre une valeur discrète (parmi plusieurs valeurs -i.e., états- définis). Un nœud reçoit des messages π (antériorité, ou support causal) de ses parents et des messages λ (probabilité, ou support de diagnostic) de ses enfants lorsque des événements sont observés. Les messages d'antériorité se propagent en descendant l'arbre et les probabilités en le remontant. Ces distributions sont discrètes, c'est pourquoi ce sont des valeurs positives dont la somme est égale à 1. La probabilité sur les nœuds terminaux ou feuilles correspond à un événement directement observable. Une table de probabilité conditionnelle (CPT) lie un enfant à un parent. Ses éléments sont donnés par :

$$CPT_{ij} = P(state=j \mid parent\_state=i)$$

En conséquence, chaque ligne d'une CPT est une distribution directe des nœuds pour un état parent particulier, c'est-à-dire :

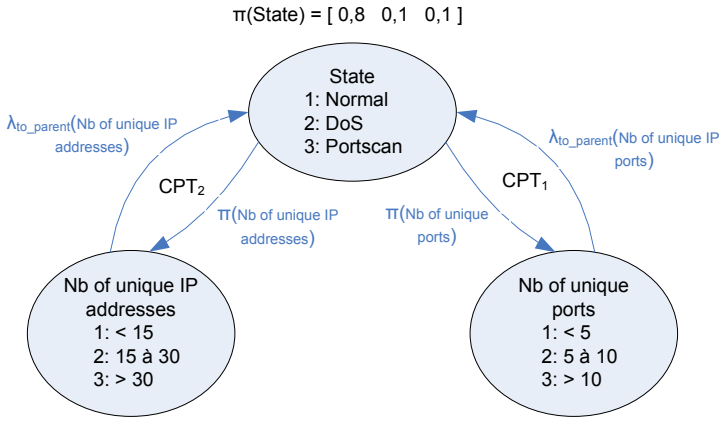
$$CPT_{ij} \geq 0, \forall i,j$$

$$\sum_j CPT_{ij} = 1, \forall j$$

[1] EMERALD (*Event Monitoring Enabling Responses to Anomalous Live Disturbances*) represents state of the art in research and development of systems and components for anomaly and misuse detection in computer systems and networks. http://www.sdl.sri.com/projects/emerald/

Les opérations de base pour la propagation de messages dans l'arbre sont exprimées sous forme de vecteur/matrice.

Considérons l'exemple suivant:



avec $CPT_1 = \begin{bmatrix} 0,9 & 0,05 & 0,05 \\ 0,2 & 0,3 & 0,5 \\ 0,05 & 0,15 & 0,8 \end{bmatrix}$ $CPT_2 = \begin{bmatrix} 0,9 & 0,05 & 0,05 \\ 0,9 & 0,05 & 0,05 \\ 0,3 & 0,3 & 0,4 \end{bmatrix}$

Un vecteur des valeurs courantes est maintenu : π(State) = [0,8 0,1 0,1].

2) π : messages envoyés des parents vers les enfants. La propagation descendante des messages π, représentés par un vecteur en ligne, est réalisée par multiplication du π parent par la CPT, c'est-à-dire :

$$\pi(node) = \alpha\pi(parent\_node) \bullet \text{CPT}$$

où α est une constante de normalisation assurant que la somme soit égale à 1.

Dans notre exemple,

$$\pi(Nb\,of\,unique\,ports) = \alpha\pi(State) \bullet CPT_1 = \alpha\begin{bmatrix}0,8 & 0,1 & 0,1\end{bmatrix} \bullet \begin{bmatrix} 0,9 & 0,05 & 0,05 \\ 0,2 & 0,3 & 0,5 \\ 0,05 & 0,15 & 0,8 \end{bmatrix}$$

$$= \alpha[0,745 \quad 0,085 \quad 0,17]$$

$$\pi(Nb\,of\,unique\,IP\,addresses) = \alpha\pi(State) \bullet CPT_2 = \alpha\begin{bmatrix}0,8 & 0,1 & 0,1\end{bmatrix} \bullet \begin{bmatrix} 0,9 & 0,05 & 0,05 \\ 0,9 & 0,05 & 0,05 \\ 0,3 & 0,3 & 0,4 \end{bmatrix}$$

$$= \alpha[0,84 \quad 0,075 \quad 0,085]$$

Le nombre d'éléments dans $\pi(node)$ peut être différent de celui dans $\pi(parent\_node)$. Il y a au plus un parent par nœud. Néanmoins il peut y avoir de multiples enfants, de sorte que la propagation ascendante des messages de probabilité nécessite une étape de fusion.

3) λ : messages envoyés des enfants vers les parents. Pour chaque nœud, le message λ, représenté par un vecteur en colonne, est propagé vers le nœud parent via la matrice de calcul :

$$\lambda\_to\_parent(node) = CPT \bullet \lambda(node)$$

Dans notre exemple, supposons l'observation d'une attaque DoS :

$$\lambda(Nb\,of\,unique\,ports) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} (i.e.\,observation1 > 10\ ports)$$

$$\lambda(Nb\,of\,unique\,IP\,addresses) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} (i.e.\,observation2 < 15\ adresses\ IP)$$

$$\lambda_{to\_parent}(Nb\,of\,unique\,ports) = CPT_1 \bullet \lambda\left(Nb\,of\,unique\,ports\right) = \begin{bmatrix} 0{,}9 & 0{,}05 & 0{,}05 \\ 0{,}2 & 0{,}3 & 0{,}5 \\ 0{,}05 & 0{,}15 & 0{,}8 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0{,}05 \\ 0{,}5 \\ 0{,}8 \end{bmatrix}$$

$$\lambda_{to\_parent}(Nb\,of\,unique\,IP\,addresses) = CPT_2 \bullet \lambda\left(Nb\,of\,unique\,IP\,addresses\right) = \begin{bmatrix} 0{,}9 & 0{,}05 & 0{,}05 \\ 0{,}9 & 0{,}05 & 0{,}05 \\ 0{,}3 & 0{,}3 & 0{,}4 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0{,}9 \\ 0{,}9 \\ 0{,}3 \end{bmatrix}$$

$\lambda(node)$ a un nombre d'éléments égal au nombre d'états dans le nœud, alors que $\lambda\_to\_parent(node)$ a un nombre d'éléments égal au nombre d'états dans le nœud parent.

4) Mise à jour de π (degré de confiance) dans le nouvel état. Les deux messages précédents sont fusionnés auprès du nœud parent via une multiplication élément par élément (produit de tous les λ_to_parent pour chaque c enfant de parent) :

$$L_i(parent) = \prod_{c \in children(parent)} \lambda\_to\_parent_i(c)$$

$$\lambda_i(parent) = L_i(parent) / \sum_j L_j(parent)$$

L représente le produit des lignes et λ est obtenu en normalisant la somme à 1.
Dans notre exemple,

$$L_i(State) = \lambda_{to\_parent_i}\left(Nb\,of\,unique\,ports\right) \times \lambda_{to\_parent_i}\left(Nb\,of\,unique\,IP\,adresses\right) = \begin{bmatrix} 0{,}05 \\ 0{,}5 \\ 0{,}8 \end{bmatrix} \bullet \begin{bmatrix} 0{,}9 \\ 0{,}9 \\ 0{,}3 \end{bmatrix} = \begin{bmatrix} 0{,}045 \\ 0{,}45 \\ 0{,}24 \end{bmatrix}$$

$$\begin{aligned} & L_1(State) = 0{,}045 \\ \Rightarrow\ & L_2(State) = 0{,}45 \\ & L_3(State) = 0{,}24 \end{aligned}$$

$$\lambda_1(State) = \frac{L_1(State)}{\sum_{j=1à3} L_j(State)} = \frac{0{,}045}{(0{,}045 + 0{,}45 + 0{,}24)} = 0{,}0612$$

$$\lambda_2(State) = \frac{L_2(State)}{\sum_{j=1à3} L_j(State)} = \frac{0{,}45}{(0{,}045 + 0{,}45 + 0{,}24)} = 0{,}6122$$

$$\lambda_3(State) = \frac{L_3(State)}{\sum_{j=1à3} L_j(State)} = \frac{0{,}24}{(0{,}045 + 0{,}45 + 0{,}24)} = 0{,}3265$$

Enfin, la croyance (*belief*) sur les états d'un nœud est obtenue par :

$$BEL_i = \beta\pi_i\lambda_i$$

où β est une constante de normalisation telle que la somme de BEL=1.

$$\Rightarrow\ \pi_{new_i}(State) = \beta \times \pi_{old_i} \times \lambda_i(State)$$

Dans notre exemple,

$$\pi_{new_1}(State) = \beta \times \pi_{old_1}(State) \times \lambda_1(State) = \beta \times 0{,}8 \times 0{,}0612 = 0{,}049\beta$$

$$\pi_{new_2}(State) = \beta \times \pi_{old_2}(State) \times \lambda_2(State) = \beta \times 0{,}1 \times 0{,}6122 = 0{,}061\beta$$

$$\pi_{new_3}(State) = \beta \times \pi_{old_3}(State) \times \lambda_3(State) = \beta \times 0{,}1 \times 0{,}3265 = 0{,}03265\beta$$

$$0{,}049\beta \times 0{,}061\beta \times 0{,}03265\beta = 1 \quad \Rightarrow \quad \beta = \frac{1}{0{,}14283} = 7$$

Donc $\pi_{new}(State) = \left[\pi_{new_1} \quad \pi_{new_2} \quad \pi_{new_3}\right] = \left[0{,}34 \quad 0{,}43 \quad 0{,}23\right]$ (tend vers le cas 2: DoS)

La Fig. 1 illustre la propagation des messages dans un fragment d'arbre.

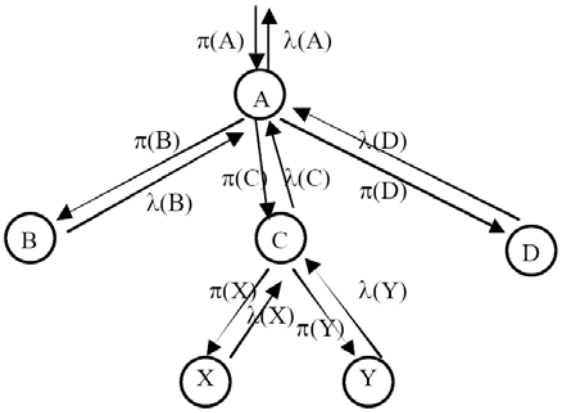


Fig. 1. Propagation des messages dans un fragment d'arbre (source: Valdes & Skinner, 2000)

Modèle de session

Le modèle de session le plus simple considère le trafic temporellement contigu d'une adresse IP particulière comme une session. Pour détecter les attaques distribuées, il considère également le trafic vers une ressource protégée, comme un serveur dans le LAN protégé. Quand un événement est observé, la liste des sessions actives est examinée à la recherche d'une correspondance. S'il y en a une, la session correspondante est mise à jour avec le nouvel événement. Sinon, une nouvelle structure de session est allouée et ses événements observables sont initialisés avec les données de l'événement. La croissance de la session est gérée par deux moyens : une opération de nettoyage (*housecleaning*) régulièrement programmée (toutes les sessions peuvent rester actives durant un certain intervalle de temps après que le dernier événement ait été vu pour la session ; quand une telle opération de nettoyage est invoquée, toutes les sessions dont le *timeout* est antérieur à l'opération de nettoyage sont désallouées) ; si la table de session est à sa taille maximale et qu'un événement est observé pour une nouvelle session, une opération de « *hindmost* » (le plus antérieur) est invoquée pour désallouer la session la plus ancienne. Le moteur d'inférence est invoqué périodiquement durant chaque session, et à chaque fois qu'une session est désallouée.

Structure d'eBayes TCP

Le nœud racine représente la classe session (inobservable) ; les nœuds enfants sont des variables observées, dérivées des données de TCPDUMP. Tous les nœuds enfants sont aussi des nœuds feuilles (observables). La structure d'eBayes TCP est

représentée par la Fig. 2 ; elle est supposée se produire à chaque intervalle d'inférence (ou période temporelle).

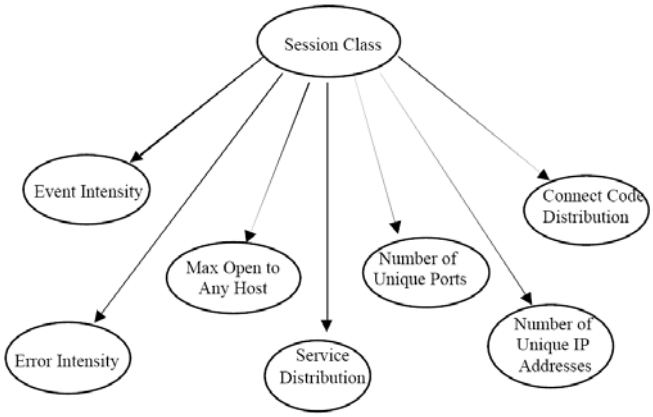


Fig. 2. eBayes TCP
(source: Valdes & Skinner, 2000)

Cette structure, parfois appelée *modèle de Bayes naïf*, suppose l'indépendance conditionnelle des enfants par rapport au parent.

Dans ce modèle, l'inférence bayésienne est appliquée pour obtenir une croyance sur des états intéressants (hypothèses) pour la session considérée (ou plus généralement, une rafale de trafic provenant d'une adresse IP donnée). Dans le modèle, les hypothèses de la classe session au nœud racine sont MAIL, HTTP, FTP, TELNET/REMOTE USAGE, OTHER NORMAL, HTTP_F (*failed http*), DICTIONARY, PROCESSTABLE, MAILBOMB, PORTSWEEP, IPSWEEP, SYNFLOOD, et OTHER ATTACKS. Les cinq premiers représentent des modes d'usage normal, les autres sont des modes d'attaque. HTTP_F décrit un modèle de sessions http longues avec des terminaisons anormales fréquemment rencontrées dans le trafic réel. A ce point, les auteurs ne considèrent pas que ces sessions représentent des attaques.

eBayes TCP est couplé avec le moniteur de disponibilité de service, qui trouve les hôtes et services valides dans le réseau protégé via un processus de découverte non supervisée. Ceci permet la détection de balayage de port furtif sans modèle prédéfini. Pour chaque session (comme déterminé par le modèle de session ci-dessus) un nombre d'éléments observables est collecté. L'inférence est réalisée périodiquement pour chaque session (la période est un nombre configurable d'événements ou de temps passé) et à chaque fois que le système croit que la session s'est terminée. Le modèle met en œuvre trois types de mesures : intensité, limite supérieure et distribution.

Une mesure d'*intensité* est un total exponentiellement atténué. Elle permet de mesurer les événements de manière pondérée par rapport au temps : un nouvel événement qui se produit augmente l'intensité de l'événement, alors que des

événements passés il y a longtemps sont pondérés pour diminuer leur importance. Les mesures d'intensité dans eBayes sont définies comme suit :

$$Event\_Intensity_{event} = e^{k\Delta t}\, Event\_Intensity_{event\text{-}1} + 1.0$$

$$Error\_Intensity_{event} = e^{k\Delta t}\, Event\_Intensity_{event\text{-}1} + (event \text{ has error return}) \; ?1.0 : 0.0$$

Δt = temps entre l'événement présent et son prédécesseur immédiat
k = constante d'affaiblissement (≤ 0)

Les mesures d'intensité ont la propriété d'avoir une croissance bornée, tant que le comportement est normal et que la constante d'atténuation est choisie de façon appropriée.. L'échelle de ces mesures est catégorisée pour obtenir les valeurs des états observés pour les nœuds respectifs.

Les mesures de *limite supérieure* suivent un total constitué d'une observation au maximum. Le système maintient une liste de ports et d'hôtes uniques accédés par la session. De cette liste est dérivé le nombre maximum de connexions ouvertes vers un hôte unique. Une option de configuration remet (ou non) ce maximum à 0 à chaque intervalle d'inférence. Le nombre total de ports et d'hôtes uniques accédés par la session sont aussi enregistrés. Comme pour les mesures d'intensité, l'échelle est catégorisée pour obtenir les états respectifs des nœuds.

Les mesures de *distribution* suivent la distribution des catégories sur une mesure discrète. Par exemple, pour chaque événement, le service est classé dans l'une des catégories MAIL, HTTP, FTP ou OTHER. Sur un intervalle d'inférence, le système maintient un total du nombre de fois où chaque service a été observé. La distribution du service est alors obtenue par :

svc_dist = {*count(MAIL) count(HTTP) count(FTP) count(REMOTE) count(OTHER)*}

Théoriquement, cette expression devrait être divisée par le nombre total pour obtenir une vraie distribution, mais la normalisation du fonctionnement interne de l'inférence bayésienne s'en charge.

Paramétrer les observations signifie paramétrer les messages λ aux nœuds enfants. La fonction `set_state` a été modifiée pour qu'elle accepte soit une valeur entière d'état observé (dans quel cas $\lambda_{obs} = 1$, $\lambda_i = 0$ pour $i \neq obs$) soit une distribution (dans quel cas $\lambda = svc\_dist$, par exemple).

Capacité d'adaptation

Le système a la capacité de s'adapter en renforçant ses modèles intégrés pour l'observation courante (en ajustant les lignes des CPTs correspondantes à l'état observé au nœud parent) ou en ajoutant un nouvel état (hypothèse) au nœud parent si l'observation courante n'est pas en concordance avec l'une des hypothèses déjà modélisées.

Ajustement adaptatif des CPTs (par renforcement)

L'adaptation par renforcement se passe comme suit. Rappelons que la CPT relie un nœud enfant à son parent. Dans la représentation des auteurs, les lignes de la CPT correspondent aux états parents, et les colonnes aux états enfants. Le système fait un total des hypothèses qui se sont révélées vraies ; il ajuste la nouvelle CPT en fonction des messages λ et des totaux. Si une seule hypothèse est dominante au nœud racine, la ligne correspondante de la matrice CPT de chaque enfant est ajustée légèrement dans la direction du message λ pour l'observation courante. Si l'hypothèse i « gagne » au nœud racine, la CPT est ajustée comme suit.

D'abord, le total effectif interne est atténué via une fonction d'atténuation :

$$counts_i^{decay} = \gamma counts_i + (1\text{-}\gamma)$$

Le comptage atténué est utilisé comme un « poids passé » pour l'ajustement, et représente le nombre effectif de fois où l'hypothèse a été récemment observée. La ligne de la CPT est d'abord convertie en un total effectif pour chaque état enfant, et l'observation courante est ajoutée comme un total additionnel distribué sur les mêmes états. Puis les éléments de la ligne sont divisés par la somme de la ligne de telle sorte que la ligne corrigée ait pour somme 1. Ceci est accompli par l'équation :

$$\mathbf{CPT}_{ij}^{adj} = (counts_i \times \mathbf{CPT}_{ij} + \lambda_j) / (\sum_j counts_i \times \mathbf{CPT}_{ij} + \lambda_j)$$

En dernier lieu, les totaux internes sont recalculés (i.e., mis à jour) pour tous les états parents :

$$counts_i = counts_i^{decay} + \begin{cases} \gamma \text{ si l'hypothèse } i \text{ est la gagnante} \\ 0 \text{ sinon} \end{cases}$$

Par cette procédure, le comptage effectif ne descend jamais en-dessous de 1 (si l'hypothèse n'est jamais observée) et ne croît jamais au-delà de $1/(1\text{-}\gamma)$ si l'hypothèse est toujours observée. Le facteur d'atténuation a été choisi de telle sorte que le total effectif varie entre 200 et 1000 observations. Les observations pour des hypothèses fréquemment rencontrées ont un ajustement CPT plus petit que les observations pour des hypothèses rares. De plus, comme seules des hypothèses « gagnantes » causent un ajustement CPT potentiel, ce système a un avantage clé sur d'autres IDS. Un grand nombre d'observations pour une hypothèse correspondant à une attaque ne sera pas considéré comme « normal », quelle que soit la fréquence de son observation, puisque son ajustement renforce seulement le modèle d'hypothèse interne d'attaque dans le système.

Génération dynamique d'hypothèses

Une autre forme d'adaptation est l'aptitude du système à générer des états. Les modèles bayésiens naïfs comme celui décrit précédemment fonctionnent bien dans la pratique comme des classifieurs et sont typiquement entraînés avec des observations pour lesquelles la vraie classe est connue. La génération dynamique d'hypothèses aborde un problème plus difficile, à savoir la situation où les données empiriques ne

sont pas étiquetées et même le nombre d'états hypothèses est inconnu. Dans cette situation, il est légitime de se demander si un système peut auto-organiser un nombre d'hypothèses qui sépare adéquatement les classes de données importantes. L'aptitude à séparer les classes d'attaques A et B l'une de l'autre est moins importante que l'aptitude à séparer A et B de l'ensemble des classes bénignes.

Pour mettre en oeuvre cette capacité, le système doit être autorisé à ajouter des hypothèses au nœud racine. Si une nouvelle observation est observée fréquemment, le système crée alors un état fictif (*dummy state*) au nœud racine (ou plus généralement, à tout nœud correctement observable), dont le total effectif est égal à 1. La distribution est uniforme sur les états enfants (chaque élément a une valeur $1/nstate_{child}$) pour cette CPT à présent. Ajouter un état se fait alors comme suit. Le mécanisme d'inférence est appliqué à une observation, et une croyance postérieure est obtenue pour l'état fictif comme s'il s'agissait d'un état normal. Si cet état temporaire (nouvelle hypothèse) « gagne » au nœud racine, il est promu à la classe d'état valide et les lignes des CPTs de tous les enfants sont modifiées via la procédure d'ajustement des CPTs décrite précédemment. Comme le total effectif de l'état fictif est 1, l'ajustement de la CPT correspond à 50% de l'observation. Puis un nouvel état fictif est ajouté, permettant au système de croître jusqu'au nombre d'états au nœud racine décrivant adéquatement les données. Cet état fictif ne doit pas être confondu avec l'hypothèse OTHER ATTACK, pour laquelle il existe un modèle initial de comportement anormal.

Il y a deux manières d'exploiter la capacité de génération des hypothèses. Dans la première, le système est initialisé avec les hypothèses normales et d'attaque décrites précédemment, à l'aide de CPTs dérivées de l'expertise dans le domaine. Le système ajuste la CPT quelque peu, mais ne choisit pas d'ajouter plus d'hypothèses si cela se passe de cette manière. Les auteurs en concluent que douze hypothèses au maximum sont nécessaires pour classifier les données. A l'autre extrême, le système a été initialisé avec une seule hypothèse valide et une hypothèse fictive au nœud racine. Un ensemble de huit données normales (bénignes) a été présenté au système, et celui-ci a généré deux états valides. Les CPTs ont été ajustées conformément à la procédure précédemment définie. Les auteurs ont alors arbitrairement décidé que tout nouvel état appris serait rapporté comme une attaque potentielle, les données présentées contenant des attaques. Le système a ajouté deux nouveaux états, qui ont capturé les attaques observées précédemment dans le modèle à onze états spécifié par un expert. Il y a eu quelques fausses alarmes, mais leur nombre est bien inférieur aux directives du Laboratoire Lincoln limitant le nombre d'alarmes à dix par jour pour une utilisation opérationnelle. C'est pourquoi, avec les capacités d'adaptation via renforcement et l'expansion de l'espace d'états décrit plus haut, il est en fait possible de lancer le système sans aucune connaissance initiale ; il organise alors un nombre approprié d'hypothèses et de valeurs CPT. De manière intéressante, ce système sépare les classes importantes (ici normal *versus* attaque) comme le modèle expert avec seulement quatre hypothèses d'état au nœud racine. Les données normales sont adéquatement représentées par deux états et la variété de données d'attaque par deux états anormaux.

Dans les étapes d'inférence intermédiaires, l'état de croyance sur la classe session passe par un modèle de transition de Markov, de façon à rapporter un état de croyance de pré-observation immédiatement avant l'étape d'inférence suivante.

Etat de transition

Par souci de simplification, les états observés pour les variables respectives sont considérés comme indépendants de ce qui a été observé pour ces variables dans les intervalles d'inférence précédents, étant donné la classe session. De même, étant donné la valeur de la classe session dans l'intervalle courant, X est indépendant de toute autre variable Y observable. En d'autres termes, pour toutes les variables observables X, Y et les intervalles d'inférence 0 à k :

$$P(X_k = x \mid Sess\_class_k = s, X_{k-1} \ldots X_0, Y_{k-1} \ldots Y_0) = P(X_k = x \mid Sess\_class_k = s)$$

L'évolution de la classe session sur les intervalles d'inférence est modélisée comme un processus de Markov à états et temps discrets. La matrice de transition est une combinaison convexe d'une matrice d'identité (pour exprimer la persistance d'état) et d'une matrice dont les lignes sont toutes égales à une distribution des messages π (*prior*) sur les valeurs possibles des classes de session (pour exprimer la tendance du processus à atténuer vers un état antérieur). En d'autres termes, pour tout $0 \leq \gamma \leq 1$, la matrice de transition **M** est donnée par :

$$\mathbf{M} = \gamma \mathbf{I} + (1-\gamma)\mathbf{P}$$

où **I** est une matrice d'identité, chaque ligne de **P** est donnée par :

$$\mathbf{P}_{i,.} = \mathbf{PRIOR}$$

et **PRIOR** est une distribution antérieure sur des valeurs possibles j pour des classes de session :

$$\mathbf{PRIOR}_j = \text{Prior probability } (Sess\_class = j)$$

$\mathbf{M}_{ij}$ est la probabilité pour que le processus soit dans l'état j au prochain événement s'il est actuellement dans l'état i. Plus généralement, si **POST_BEL** est l'état courant de croyance (une distribution sur les valeurs d'état possibles, étant donné l'observation jusqu'à cet intervalle temporel inclus), la multiplication gauche avec **M** redistribue la croyance pour obtenir la croyance antérieure à l'observation définie par :

$$\mathbf{PRE\_BEL}_k = \mathbf{POST\_BEL}_{k-1}\mathbf{M}$$

Le paramètre γ est modifié pour capturer, quoiqu'imparfaitement, la nature continue du processus sous-jacent. Typiquement, la fonction d'inférence est invoquée tous les cent événements à l'intérieur d'une session et à chaque fois que la session entre dans un état inactif. Certaines sessions comportent moins de cent événements au total, alors que d'autres (particulièrement lors d'attaques de déni de service) comptent des dizaines de milliers d'événements dans un très court intervalle de temps. Dans ce dernier cas, bien que de nombreuses étapes d'inférence soient invoquées, les auteurs préfèrent conserver un paramètre de persistance moyennement élevé (de l'ordre de 0.75) car très peu de temps s'est écoulé. Si le paramètre est 0, la croyance renvoie à π (*prior*) à chaque événement.

Il peut être montré que, hormis si γ=1, la multiplication itérative de **M** par elle-même résulte en une matrice qui approche P, c'est-à-dire :

$$\lim_{n\to\infty}\mathbf{M}^n = \mathbf{P}$$

Dans la pratique, cette limite est presque atteinte pour des valeurs petites de *n*. Le résultat de cette observation est intéressant du point de vue intuitif : en l'absence d'une preuve de renforcement d'événements subséquents, la distribution de croyance a tendance à renvoyer à π (*prior*).

L'opération d'inférence à l'intervalle *k* commence par fixer le message bayésien π à $\mathbf{PRE\_BEL}_k$. Puis les événements observables sur l'intervalle sont soumis aux nœuds feuilles, et l'état de croyance au nœud racine est extrait. Si cet état est considéré comme suffisamment suspect, un message d'alerte est écrit à la fois vers un log d'alerte et dans le format IDIP.

Résultats

Avantages : le modèle a été testé avec les données TCPDUMP du Laboratoire Lincoln IDEVAL. Il s'est montré très efficace contre les attaques par diffusion (*flood*) et les attaques par balayage non furtif, et moyennement efficace contre les attaques par balayage furtif.

Désavantages : le trafic de l'extérieur vers l'intérieur a été examiné à l'aide du protocole TCP/IP. Ceci signifie que les attaques de console, les attaques provenant de l'intérieur et les attaques exploitant d'autres protocoles (tels qu'IDP et UDP) sont invisibles. Ceci limite le système aux attaques utilisant TCP/IP.

Les attaques détectées au Laboratoire Lincoln sont les suivantes :

- attaque de Satan (balayage de ports) et attaque TCPRESET détectées comme balayage de ports ;
- attaques MAILBOMB et PROCESSTABLE détectées à 100% ;
- attaques sur les mots de passe détectées comme OTHER ou DICTIONARY ;

Les attaques par balayage de port/IP et déni de service ont été expérimentées sur données réelles.

Utilité de l'apprentissage

L'apprentissage est utile à la fois dans le raffinement d'hypothèses existantes, et pour développer de nouvelles hypothèses relatives à des modalités normales et d'attaque. Cependant, les auteurs ont observé un meilleur fonctionnement du système si sa capacité d'adaptation est désactivée, ceci pour plusieurs raisons. En premier lieu, les attaques et événements nécessitant une alerte ne constituent qu'une très petite fraction du trafic dans le monde réel, de telle sorte que l'apprentissage d'une modalité d'attaque qui ne sera peut-être vue qu'une seule fois est problématique. En second lieu, les hypothèses normales se sont « durcies » par la capacité d'adaptation jusqu'à devenir relativement intolérantes à des sorties erronées. La proportion de telles sorties pour des raisons non malveillantes est trop élevée pour être intolérable du point de vue des alertes, mais est trop faible pour laisser suffisamment d' « espace d'action » si l'adaptation est permise indéfiniment. Pour le moment, les auteurs utilisent donc le système en mode adaptatif pour identifier les modalités sur lesquels il n'est pas

possible d'anticiper et les déviations importantes de ce qui est observé dans le trafic réel. Les résultats de cette analyse sont alors modérés par un jugement humain, aboutissant à une spécification de groupe pour les CPTs. Les auteurs vérifient ensuite que ce nouvel encodage reste sensible à des ensembles de données simulées. Aves ce dispositif, la plupart des attaques détectées jusqu'alors dans les données de Lincoln sont détectées, et les événements devant générer une alerte dans les données réelles sont détectées avec un niveau acceptable de fausses alertes générées.

Conclusion

Cet article décrit les capacités du système de détection eBayes, qui combine inférence bayésienne et modèles de transition entre les inférences pour déterminer si une rafale particulière de trafic contient une attaque. Un composant couplé surveille la disponibilité des services valides, eux-mêmes appris via un processus de découverte non supervisée.

L'efficacité de ce système a été démontrée d'une part par les résultats des données du laboratoire de détection d'intrusion Lincoln, et d'autre part par une expérimentation sur un site réel. Ceci fournit plusieurs nouvelles conclusions importantes :

- L'encodage probabiliste des modèles d'attaque fournit une capacité complémentaire pour la détection d'anomalie et l'analyse de signature, retenant le potentiel de généralisation de la première et la sensitivité/spécificité de la seconde.
- Des attaques distribuées sont à présent potentiellement détectées, dans lesquelles aucune des sessions d'attaque n'est individuellement suffisamment suspecte pour générer une alerte. Ceci est réalisé via une corrélation par agrégation.
- Une fois qu'un déni de service a réussi, le système est moins enclin à générer de fausses alertes pour des clients non malveillants demandant le service durant l'attaque (ces clients sont considérés comme « dommage collatéral »). Cette forme de corrélation fusionne la croyance d'une attaque en cours avec le symptôme de l'attaque (le service est désactivé lorsque l'attaque atteint ses objectifs) pour expliquer les alertes découlant de sessions de « dommage collatéral ». En tant que tel, le système corrélant symptômes et attaques fournit une réduction effective des fausses alarmes, mais fournit toujours à l'administrateur une alerte pour l'attaque initiale ainsi qu'une indication sur le statut de la victime hôte/port.

Les auteurs utilisent ce système avec un moniteur de session TCP sur leur propre passerelle TCP. Alors qu'ils ne disposent pas de connaissance de base pour ce trafic, des attaques de balayage et des activités de *spidering* sont régulièrement identifiées, ainsi que d'occasionnelles tentatives de déni de service. Des défaillances et rétablissements de services sont également détectés pour ce qui paraît être des défaillances non malveillantes.