Institut
Mines-Télécom

RoCaWeb : Choix algorithmiques et Questions d'implémentation

Djibrilla Amadou Kountché

Agenda

Introduction

Version courante de RoCaWeb

- Alignement de séquences ;
- Génération d'expressions régulières, Typage ;
- Choix de l'algorithme par validation croisée ;
- IHM, Reverse Proxy.

Vers la V3

- Sources de données : Outils Big Data ;
- Nouveaux algorithmes ;
- Adaptation du Reverse Proxy (ModSecurity) ;
- Mise en place d'un environnement de test et autres améliorations.

Gestion des données

Sources

- Logs du serveur web (CLF) ;
- Logs du reverse proxy ;
- D'autres entrées à prendre en compte ?

Formats et Plate-formes

- Formats : XML, JSON, CSV, Tableurs, etc.
- Plate-formes : Apache Hadoop, Apache Sparks, SpringXD, etc. (technos en rapide évolution) ;
- Apache Mahout pour l'apprentissage.

Bases de données

- NoSQL : MongoDB, CouchDB ?
- Choix de ELK.

Choix de ModSecurity

- Reverse proxy mature ;
- Extension Lua ;
- Livraison d'une distribution avec les librairies Lua.
  - librairies de calcul scientifique. Ex : Torch, SciLua

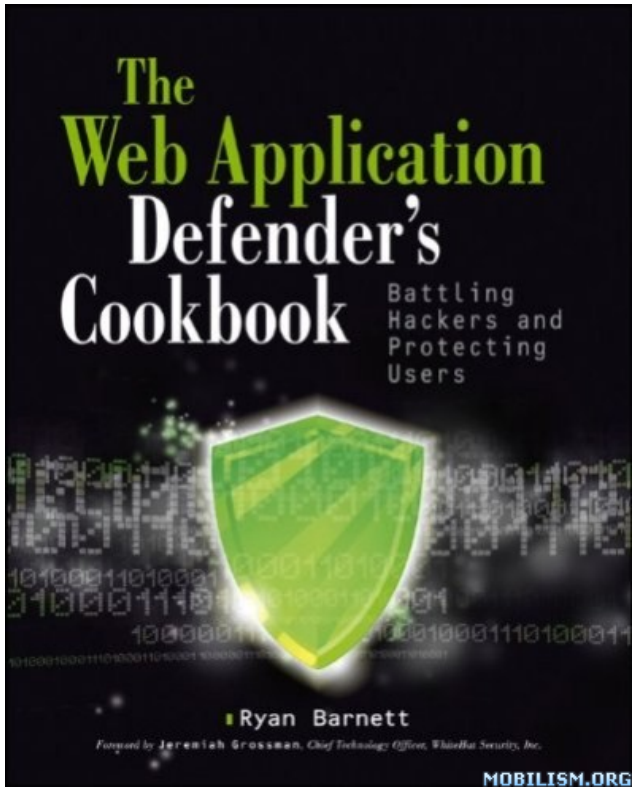
Environnement de test

- Mise en production de RoCaWeb ;
- Récolter des données sur les performances des algorithmes et du reverse proxy ;
  - Sur chaque méthode ;
  - Sur la combinaison (série ou parallèle) des méthodes.
- Valider les hypothèses ;
- Retenir les méthodes les plus efficaces ou émettre des recommandations.

Caractérisation de l'apprentissage

Définir et prendre en compte les caractéristiques d'un comportement sain (site et utilisateur) ;

- Application ;
- Utilisateur ;
- Requête ;
- Réponse ;
- Paramètre ;
- Réapprentissage.



Application

- Sous-applications ;
- Architecture ;
  - Cas de Siebel
  - Site d'une seule page
  - Traitement des cas particuliers
- Requêtes et Réponses ;
- Utilisateurs ;

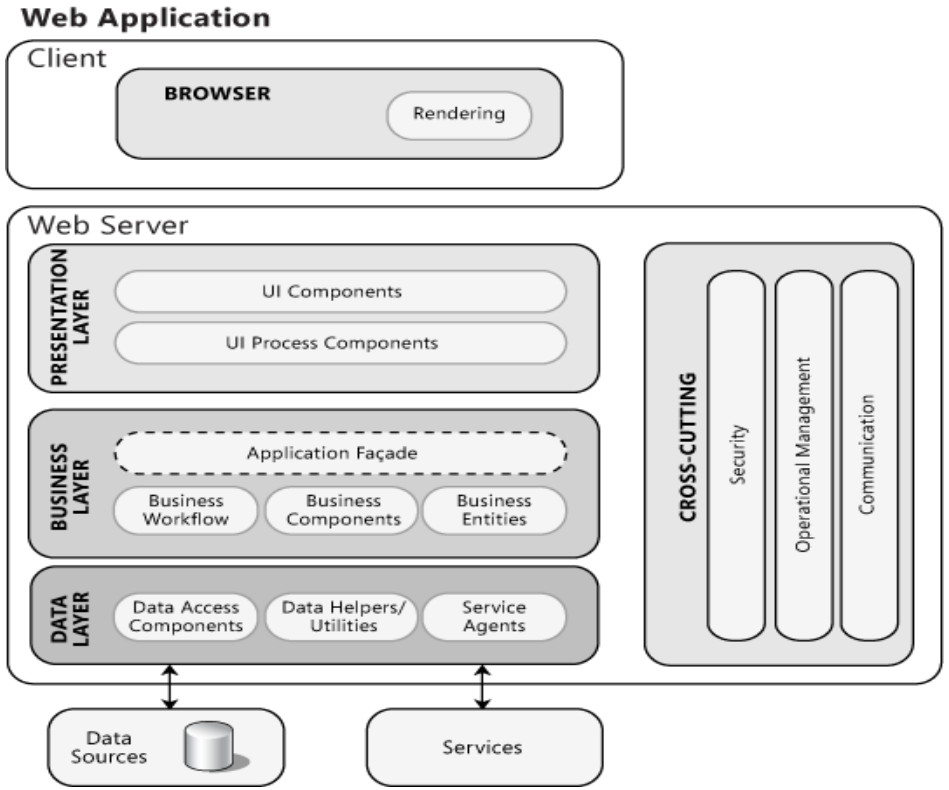


FIGURE – Source msdn.microsoft.com

Requête/Réponse

- En-tête et corps ;
- Nombre de paramètres (intervalle min/max) ;
- Noms des paramètres ;
- Longueurs des paramètres (intervalle min/max) ;
- Types des paramètres ;
- Pair requête/réponse ;
- Payload ?

Traitement des réponses

Prévu dans la version 4.

Utilisateur

- Requête ;
- Session ;
- Adresse IP ;
- etc.

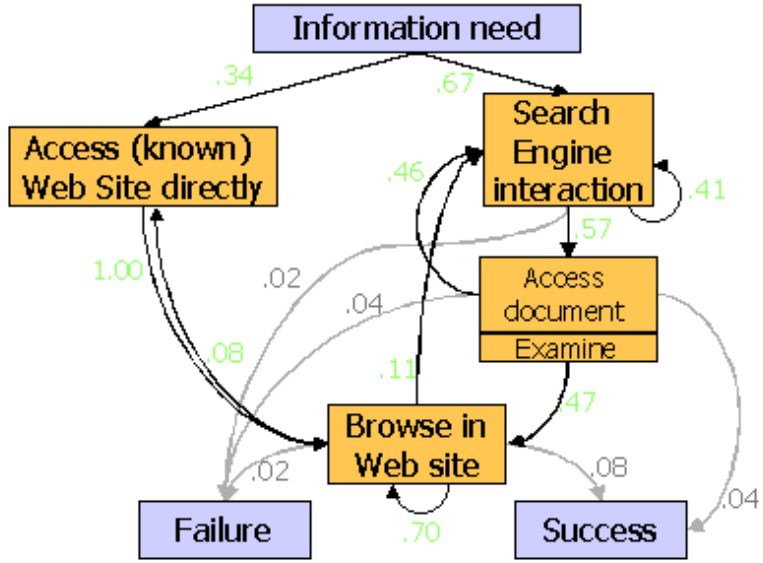


FIGURE – Exemple de modélisation de comportements d'utilisateurs. Source[?]

Paramètre [?]

- Longueur ;
- Type :
  - Drapeau
  - Digits
  - Alphabétique
  - Alphanumérique
  - Émail
  - Chemin
  - URL
  - $SafeText$
  - Énumération

Sommaire

Les algorithmes

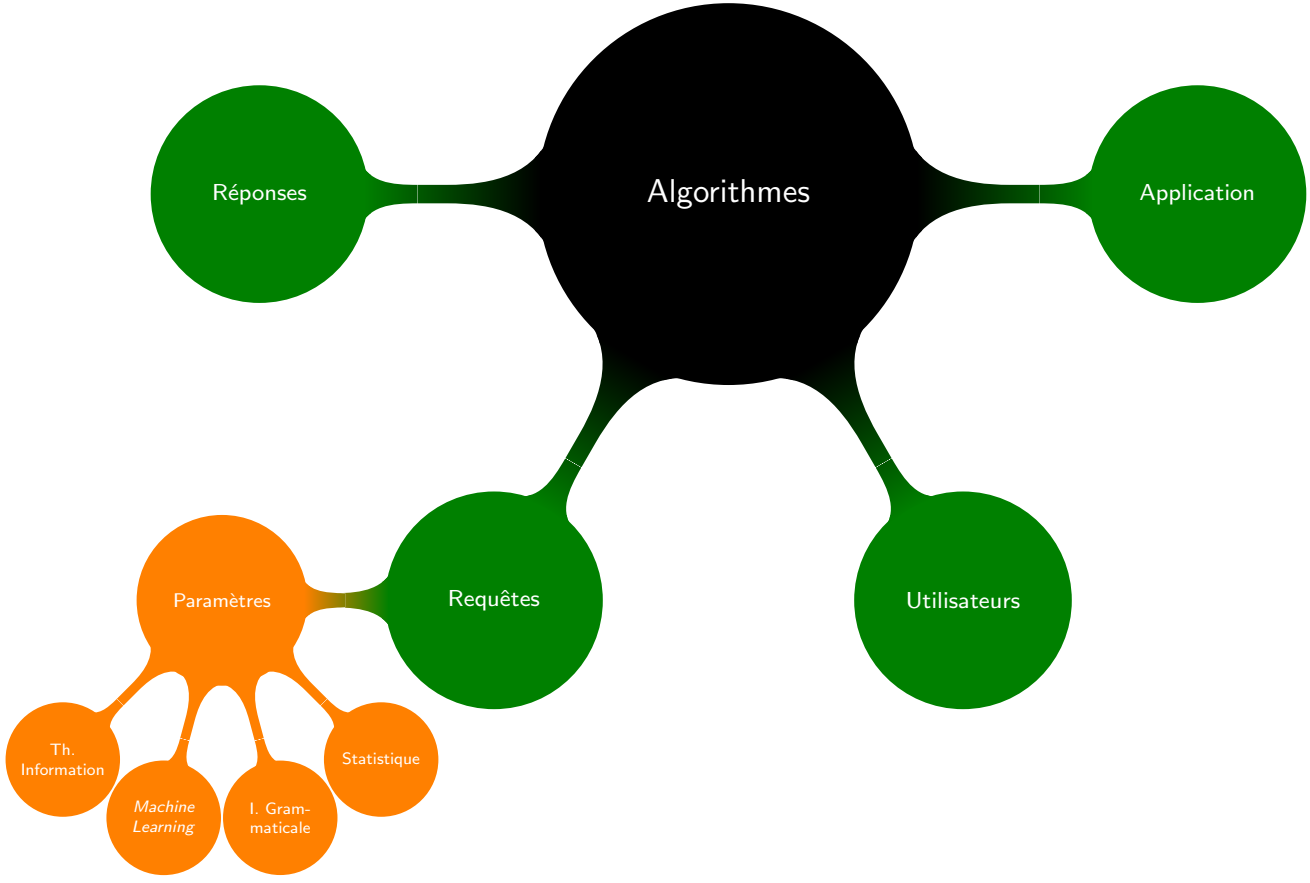


FIGURE – Mindmap des algorithmes.

Les méthodes statistiques

Hypothèse

- Données ont été générées par un modèle stochastique ;
- Données normales : zones de plus forte probabilité du modèle ;
- Données anormales : zones de faible probabilité.

Essaient d'estimer les paramètres du modèle.

Paramétriques

- Font une hypothèse sur le modèle et estiment ses paramètres à partir des données ;

Non-paramétriques

- Pas d'hypothèse sur les paramètres.

Méthode de Chebychev appliquée à la longueur

Hypothèses[?]

- Les longueurs des valeurs du paramètre n'évoluent pas énormément entre les requêtes.
- Exemple : certains champs d'un formulaire ou $token$ à taille fixe (identifiant de session)

Théorème

Soit une variable aléatoire X avec $Var[X] < +\infty$. Alors, pour tout $t > 0$, l'inégalité suivante tiens [**?**] :

$$p(|X - E[X]| \geq t \times \sigma_X) \leq \frac{1}{t^2} \quad (1)$$

L'inégalité de Chebychev ne requiert que la connaissance de la moyenne et la variance.

Méthode de Chebychev appliquée à la longueur

Phase d'apprentissage

Soient :

- A un attribut d'une requête ;
- $A = \{a_i, i = 1...n\}$ valeurs collectées

Déterminer :

- μ : la moyenne des longueurs des a_i ;
- σ : la variance ;

Méthode de Chebychev appliquée à la longueur

Phase de détection

Soient :

- a_k une valeur à évaluer ;
- μ, σ : les valeurs déterminées précédemment.

Variante :

$$p(|X-\mu| > |l-\mu|) < p(l) = \begin{cases} \frac{\sigma^2}{(l-\mu)^2} & \text{Si } l \geq \mu \\ 1 & \text{Sinon} \end{cases} \quad (2)$$

l est la longueur courante. Retourne $p(l)$

Exemples

- Données pour le paramètre action : {login, edit_event, delete_event, add_event, users, logs, logout } ;
- longueurs : $\{5, 10, 12, 9, 5, 4, 6\}$;
- $\mu = 7.28$
- $\sigma = 2.81$

Méthode de Chebychev : Avantages et Inconvénients

Avantages

- Efficace dans la détection des valeurs abbérantes ;
- Simplicité d'implémentation ;
- Complexité linéaire.

Inconvénients

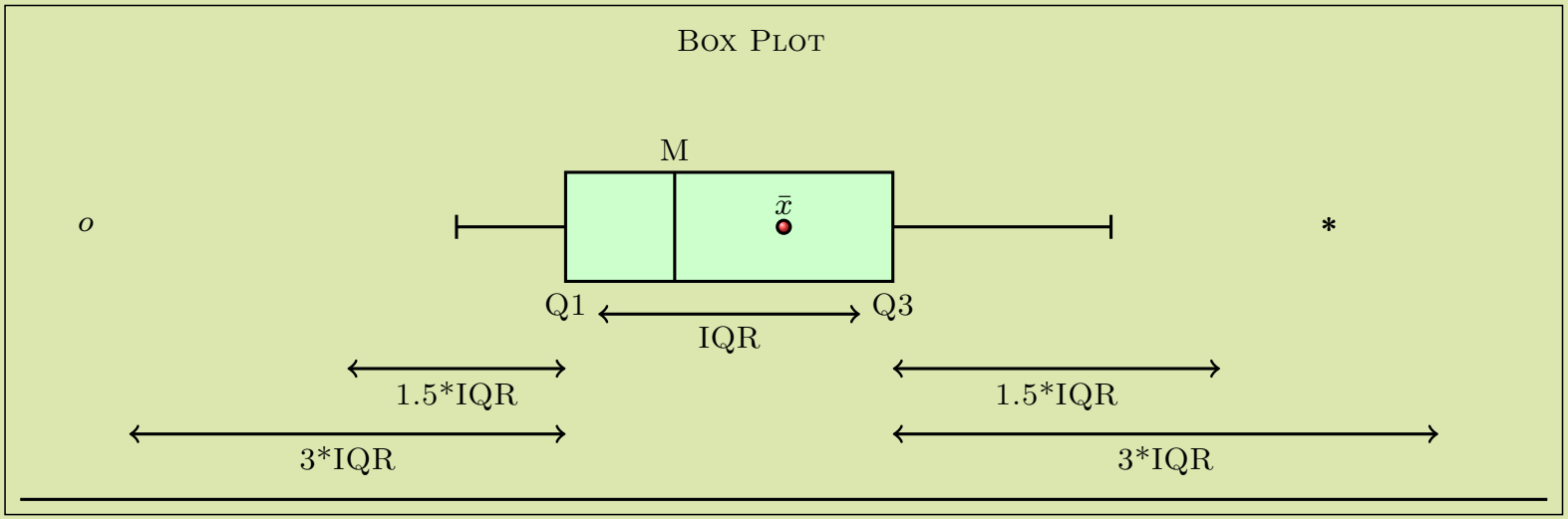
- Ne rend pas compte la structure ;

D'autres méthodes liées à la longueur

Détection de valeurs abbérantes

- Contrôle de qualité : $|length - mean| \leq 3 \times std$;

Blox Plot



Distribution des caractères de l'attribut

Hypothèses

- Les attributs ont une structure régulière ;
- Les attributs peuvent être lues par des humains ;
- Contiennent presque toujours des caractères imprimables ;

Distribution des caractères de l'attribut

Phase d'apprentissage

- Déterminer la distribution de référence ou ICD ;

Soient :

- $A = \{a_1, a_2, ..., a_n\}$;
- Σ un alphabet ;
- a_i sont définis sur Σ^*.

Une distribution de caractères est définie par :

$$CD = \{n_i, i = 1..k\} \tag{3}$$

Distribution des caractères de l'attribut

Distribution de référence

- Pour chaque paramètre déterminer le CD
- Trier décroissant CD
- $ICD = \{f_i = n_i/k, i = 1...k\}$
- Pour $k = 256$, tous les caractères possibles [**?**].

Phase de détection

Soient :

- le CD d'une valeur d'un paramètre
- et l'ICD de toutes les valeurs observées

Déterminer la probabilité en utilisant un test de χ^2

Distribution des caractères de l'attribut

Test de χ^2

$$\begin{cases} H_0 & : \text{La distribution de caractères provient de l'ICD} \\ H_1 & : \text{La distribution ne provient pas de l'ICD.} \end{cases}$$

1. Fixer le nombre de "bins". Six "bins" : $0, 1-3, 4-6, 7-11, 12-15, 16-255$
2. Calculer :

$$\chi^2 = \sum_{i=1} \frac{(O_i - E_i)^2}{E_i} \tag{4}$$

   Où :

   - O_i les valeurs observés ;
   - $E_i = f_i \times length(a_k)$.

3. Retourner la $p - value$, en fonction du nombre de degrés de liberté.

Distribution des caractères de l'attribut : Exemple

Données

$\{login, edit\_event, delete\_event, add\_event, users, logs, logout\}$

- login $= \{1, 1, 1, 1, 1\} \rightarrow \{0.2, 0.2, 0.2, 0.2, 0.2\}$
- edit_event $= \{3, 1, 1, 2, 1, 1, 1\} \rightarrow \{0.3, 0.1, 0.1, 0.2, 0.1, 0.1, 0.1\}$
- etc.

Modèle apprise

Démonstration.

Distribution des caractères de l'attribut

Avantages

- détection des très longues valeurs ;

Inconvénients

- Choix du nombre de bins influence la $p - value$
- D'autres problèmes liés à la pertinence du test de χ^2

Inférence de la structure

Hypothèses

- Grammaire régulière (inconnue) utilisée pour générer les valeurs ;
- Grammaire probabiliste : assigne une probabilité à chaque production.

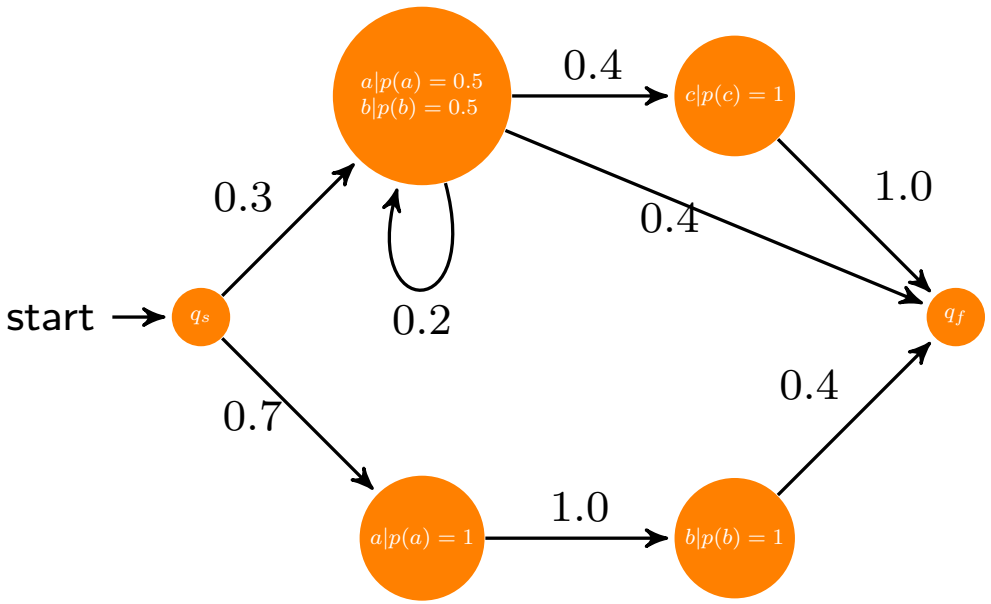


FIGURE – Exemple de grammaire probabiliste

Inférence de la structure : Modèle de Markov

Hypothèses de Markov

Soient :

- $Y_i, i = 1, \cdots, T$ les observations séquentielles
- $S_i, i = 1, \cdots, T$ les états cachés

Hypothèses :

1. Observations Y_i générées à l'instant t par un processus aux états cachés à l'observateur
2. États cachés satisfont à la propriété de Markov : à l'instant t, il suffit de connaître l'état à $t - 1^*$.

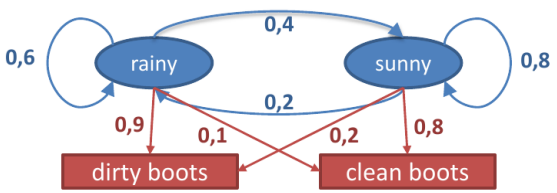


FIGURE – HMM du prisonnier en confinement

Inférence de la structure

Phase d'apprentissage

- Utilisant un réseau bayésien :

$$p(Modele|Donnees) = \frac{p(Donnees|Modele) \times p(Modele)}{p(Donnees)} \quad (5)$$

  Où :
  - $p(Donnees|Modele)$ est calculée suivant l'équation 6 ;
- L'apprentissage des probabilités est faite par la méthode de Stolcke et Omohundro[**?**, **?**].
- Illustrations

Inférence de la structure

Phase de détection

Soient :

- M : le modèle de Markov appris précédemment ;
- $w = \{w_1, w_2, \cdots, w_n\}$: une valeur du paramètres.

Déterminer la probabilité $p(w)$ selon :

$$p(w) = p(w_1, w_2, \cdots, w_n) = \sum_{(chemin)} \prod_{(etats)} p_{q_i}(w_i) \times p(t_i) \tag{6}$$

Retourne :

$$p(w) = \begin{cases} 1 & \text{Si le mot est une sortie valide de modèle} \\ 0 & \text{Sinon} \end{cases} \tag{7}$$

Construction d'énumération

Hypothèses

- Les données proviennent d'un ensemble discret et fini (dictionnaire ou énumération) ;
- Il est possible d'apprendre l'ensemble à partir des valeurs observées.

Phase d'apprentissage

Consiste à déterminer si les valeurs sont liées par un seuil t. Si c'est le cas, on admet qu'elles sont une énumération.
Soient deux fonctions :

- $f(x) = x$
- $g(x) = \begin{cases} g(x-1)+1 & \text{Si la valeur est nouvelle} \\ g(x-1)-1 & \text{Si la valeur a déjà été observée} \\ 0 & \text{Si } x = 0 \end{cases}$
- x défini sur N_0

Construction d'énumération

Phase d'apprentissage

Déterminer le coefficient de corrélation :

$$\rho = \frac{covar(f,g)}{\sqrt{var(f) \times var(g)}} \tag{8}$$

Si : $\begin{cases} \rho < 0 & \text{Considérer une énumération} \\ \rho \geq 0 & \text{Aléatoire} \end{cases}$

Construire l'énumération E.

Phase de détection

Soient :

- l'énumération et une valeur w
- retourner $p(w) = \begin{cases} 1 & \text{Si } w \in E \\ 0 & \text{Sinon} \end{cases}$

Construction d'énumération : Exemple

- Pays = {France, Allemagne, Autriche, Allemagne, Niger, France, France, Chili, Allemagne, Niger, France}
- Dérouler.

Construction d'énumération

- Avantages : facilité d'implémentation et automatisation de la définition de dictionnaire.
- Inconvénients : nécessite un grand nombre de données
- Effectuer un test statistiques (souvent biais d'échantillonnage) ;

Présence ou Absence d'un attribut

Hypothèse

- Absence ou la présence d'un ou de plusieurs paramètres (mutuellement exclusifs) dans une requêtes indiquerait une anomalie
- Détection des comportement anormaux consistant à envoyer des requêtes au hasard

Phase d'apprentissage

Pour chaque requête enregistrer toutes les attributs. $A = \{a_1, a_2, \cdots, a_n\}$
$S = \{A_{q_1}, A_{q_n}, \cdots, A_{q_n}\}$

Phase de détection

Soient :

- l'ensemble S
- une requête q.
- Retourner : $p(q) = \begin{cases} 1 & \text{Si tous les attributs sont présents} \\ 0 & \text{Sinon} \end{cases}$

Ordre des attributs

Hypothèses

- Les requêtes légitimes contiennent souvent les mêmes attributs dans le même ordre ;
- L'ordre relatif des attributs est préservé même si certains sont omis dans la requête.

Phase d'apprentissage

- Déterminer les contraintes d'ordre entre tous les k attributs.
- Un attribut a_i précède un autre a_j si :
  - a_i et a_j apparaissent dans la même requête ;
  - et a_i vient avant a_j dans la liste ordonnées des attributs de toutes les requêtes où ils apparaissent ensemble.
- L'ensemble O des contraintes est défini par :
  $O = \{(a_i, a_j) : a_i \text{ précède } a_j \text{ et } a_i, a_j \in (S_{q_j} : j = 1, \ldots, n)\}$

Ordre des attributs

Phase d'apprentissage

- O_i est défini comme un graphe $G(V, E)$
- $V = \{a_i, i = 1, \ldots, n\}$
- E pour chaque requête q_j avec un ensemble ordonnée de requêtes, pour chaque pair (a_i, a_j) un chemin orienté est introduit du nœud n_i vers n_j.
- Le graphe contient, à la fin, toutes les contraintes d'ordre imposées par les données.
- Utiliser l'algorithme de Tarjan [**?**] pour supprimer les cycles.

Ordre des attributs

Phase de détection

Soient :

- Le graphe G
- une requêtes q

Analyser toutes les paires (a_i, a_j) avec $i \neq j, 1 \leq j \leq i$ pour détecter de potentielles violations survenues.
Une violation correspond à une paire (a_j, a_i) avec $(a_i, a_j) \in O$

Retourner : $p(q) = \begin{cases} 1 & \text{Si les contraintes sont respectées} \\ 0 & \text{Si les attributs ont été alterné} \end{cases}$

Illustrations :

- p,q,r,s
- p,q,s,t

Fréquence d'accès

Hypothèse

- Les patterns de fréquence d'accès à un site sont relativement constant.

Phase d'apprentissage

- Deux types de fréquences :
  - La fréquence d'accès à partir d'une adresse IP ;
  - La fréquence totale pour toutes les adresses IP.
- Enregistrer :
  - Les temps de la première et de la dernière requête ;
  - Diviser ce temps en intervalle de $10s$;
  - Déterminer les deux types de fréquences ;
  - Les deux fréquences sont modélisés comme variable aléatoire ;
  - Déterminer (μ, σ) pour X et Y .

Fréquence d'accès

Phase de détection

Soient :

- Les variables aléatoires X, Y ;
- Et les deux types de fréquences (x, y) pour une requête.

Déterminer la probabilité de Chebychev pour chacune.

Retoruner $p = \frac{p_x + p_y}{2}$

Délais entre requête

Hypothèse

Les délais entre requêtes d'un utilisateur normal ont une grande variance.

Phase d'apprentissage

- Déterminer la distribution des temps entre les requêtes au niveau de l'application ;
- Pour chaque client, sauvegarder ce temps ;
- Regrouper ces temps en bins.

Phase de détection

- Appliquer un test de χ^2

Ordre d'invocation

Hypothèse

L'ordre d'accès aux sous-application d'un site peut être modélisé par un modèle de Markov.

Phase de détection

Voir Inference de la structure.

Ordre d'invocation

Phase d'apprentissage

- Regrouper toutes les invocations des programmes selon l'adresse IP source de la requête ;
- Identifier les sessions (s) (liste d'invocation de programmes) : $s =< chemin_1, \cdots, chemin_n >$
- Les invocations d'une même session sont déterminées par une contrainte (inter-arrival time) (les invocation proches dans le temps sont considérés comme de la même session)
- La session est ensuite traduite en chaîne de caractères ;
- Apprendre le modèle de Markov comme pour l'inférence.

Méthodes de classification

Hypothèse

- Un classifieur pouvant distinguer entre comportement normal et anormal peut être apprise à partir des données.

Type

1. Une classe ;
2. Plusieurs classes.

Mise en œuvre prévue des algorithmes :

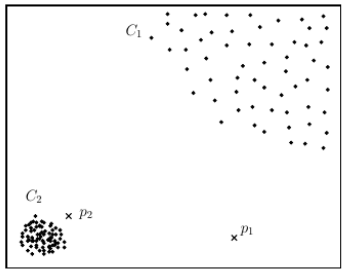
- Réseau de neurones ;
- SVM ;

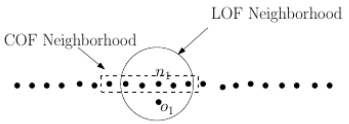
Méthode de voisinage

Hypothèse

Les données normales sont dans des zones de fortes densités ;

- KNN ;
- Local outlier Factor : ratio de la densité de kpv et de la densité autour de la données.
- Connexity-based Local Factor ;





Clustering

Hypothèse

Les données normales peuvent être regroupé en cluster.

- DBSCAN ;
- K-means ;
- etc.

Théorie de l'information

Hypothèse

L'anomalie induit des irrégularités dans la quantité d'information dans les données.

Apprentissage

Soient :

- D une base de données ;
- $C(D)$ la complexité de D.

Consiste à déterminer le sous-ensemble, I, de D tel que :

$$\max(C(D) - C(D - I)) \tag{9}$$

Mesure de la complexité :

- Complexité de Kolmogorov (ex. taille de la base compressé) ;
- Entropie, incertitude relative ;

Récapitulatif

Algorithmes	Implémentation	Règle MS
Chebychev	$\surd$	$\surd$
χ^2	$\surd$	$\surd$
Inférence Gram.	En cours	x
Construction enum.	$\surd$	x
Présence Absence	x	x
Fréquence d'accès	$\surd$	x
Ordre Invocation	En cours	x
Méthode de Classification	x	x
Autres	x	x

Sommaire

Implémentation de la validation en Lua

- Phase 1 : Implémentation des algorithmes d'apprentissage en Java
- Phase 2 : Formatage des règles au format ModSecurity ;
- Phase 3 : Implémentation de validateur en Lua.

Cas de figures pour les modèles :

- Exporter directement les paramètres du modèle dans la règle ;
- Sérialiser le modèle en XML, JSON, etc. Puis reconstruire le modèle en Lua.

Les modes de validation

Constat

- Pas de classifieurs universelles ;
- Classifieur peut ne pas discriminer des classes ;
- Réglage est difficile ;
- Importance des choix initiaux ;

Pourquoi combiner ?

- Efficacité
- Précision
- Architecture :
  - Série ;
  - Parallèle (Score) ;
  - Hybride ;
- Implémentation.

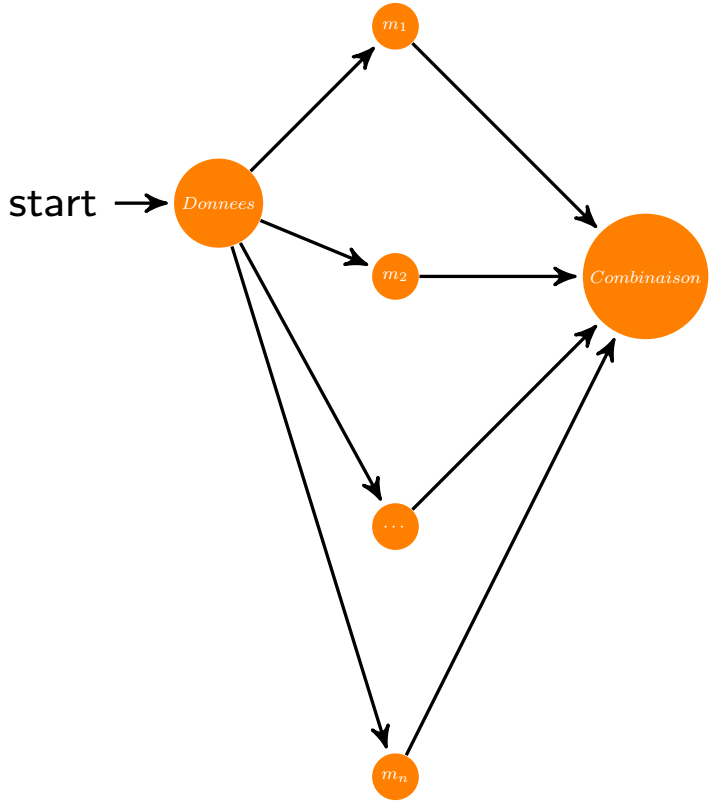
Intérêts

- Distribuer les caractéristiques ;
- Exploiter la complémentarité ;
- Prendre en compte les performances divers ;

Validation en parallèle

Description

- Classifieurs opèrent indépendamment ;
- Recherche de consensus ;
- Facile à mettre en œuvre ;
- Activation de tous les classifieurs



Validation en parallèle : Score

Définition

Utilisée par Kruegel et al [?]. Soient :

- w_m le poids du modèle ou algorithme m ;
- p_m : la probabilité retournée par le modèle.

Le score d'anomalie est calculé par :

$$score = \sum_{m \in Models} w_m * (1 - p_m) \tag{10}$$

Disponible dans ModSecurity avec la variable TX_ANOMALY_SCORE

Validation en série

Description

- Niveaux successif de décision réduisant progressivement le nombre de classes ;
- Un classifieur par niveau prenant en compte (rejets et décision précédents) ;
- Filtrage progressif des décisions (réduction de l'ambiguïté)
- Sensible à l'ordre ;
- Connaissance a priori ;
- Dépend de l'application.

Sommaire

Conclusions

- Dans la V3, un algorithme ou plusieurs de chaque catégorie ;
- Implémenter des critères d'évaluer des méthodes et lien avec la détection ;
- Intégrer plusieurs technos (ModSecurity, ELK, apprentissage) ;
- Travaux vers la V4.

Questions

Vos questions ? ? ?

Sommaire

Bibliographie I