LogAnalyzer : monitoring adaptatif d'un flux de données

T. Guyet[1,2] R. Quiniou[2] M.-O. Cordier[3]

[1] AGROCAMPUS-OUEST
[2] INRIA - IRISA
[3] Université Rennes-1 - IRISA

65 rue de Saint Brieuc, CS 84215, 35 042 Rennes Cedex, FRANCE
thomas.guyet@agrocampus-ouest.fr

Résumé

LogAnalyzer est un logiciel pour la détection adaptative des dysfonctionnements d'un système à partir d'un flux d'alarmes. Le but d'un système adaptatif de détection de malfonction est de monitorer le système tout en s'adaptant à son évolution. LogAnalyzer s'appuie sur une approche de détection utilisant plusieurs diagnostiqueurs et un méta-diagnostiqueur. Le logiciel propose deux modes d'analyse : un mode d'analyse exploratoire et interactif pour l'aide à la conception de diagnostiqueurs et un mode en ligne pour l'évaluation d'une méthode de détection sur de grands jeux de données. LogAnalyzer est évalué sur des flux de requêtes à un serveur HTTP pour une détection adaptative d'intrusions.

Mots Clef

Flux de données, meta-diagnostic, ensemble de diagnostiqueurs, détection d'intrusions

1 Monitoring adaptatif

Le monitoring adaptatif consiste à surveiller un système tout en adaptant la surveillance aux changements du fonctionnement système liés à des phénomènes non-observables (*Concept drift* [3]). De manière maintenant classique, la surveillance utilise un modèle, statistique ou symbolique, du comportement du système surveillé. Les observations réalisées sur le système surveillé et le modèle servent à déterminer si le comportement du système est normal ou anormal. L'adaptation de la surveillance passe par l'adaptation dynamique des modèles qui servent à effectuer le diagnostic du système. Une manière de la réaliser est de concevoir un méta-diagnostiqueur capable de diagnotiquer l'état du diagnostiqueur (*cf.* Figure 1). Le monitoring adaptatif par un méta-diagnotiqueur soulève trois problèmes majeurs :

1. la détection d'un besoin d'adaptation du modèle,
2. l'identification de la partie du modèle à adapter,
3. la détermination (non-supervisée) du diagnostic qu'il faudrait obtenir.

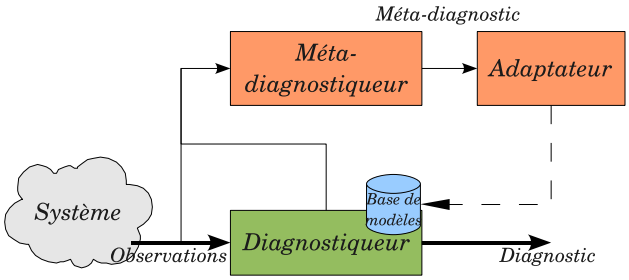

FIG. 1 – Vue d'ensemble du fonctionnement du diagnostic et du méta-diagnostiqueur.

Nous proposons une méthode de diagnostic multi-sources qui repose sur un ensemble de diagnostiqueurs, et des contraintes d'intégrité sur les résultats des diagnostiqueurs exprimées *a priori* par leur concepteur [1]. Les diagnostiqueurs utilisent des sources d'informations et des méthodes de diagnostic qui leur sont propres. Chaque diagnostiqueur construit un diagnostic *local* de l'état du système à un moment donné. Le diagnostic *global* est obtenu par fusion de tous les diagnostics fournis par les diagnostiqueurs à un moment donné. Les inter-dépendances et les redondances connues entre les sources d'information permettent au concepteur de formuler des contraintes d'intégrité sur les résultats des diagnostiqueurs et s'exprimant, par exemple, comme une fonction liant les prédictions de deux diagnostiqueurs (e.g. l'*identité*).
La détection d'un besoin d'adaptation est réalisée par la vérification des contraintes d'intégrité. Si une des contraintes n'est pas satisfaite lors d'un diagnostic, le méta-diagnostiqueur détecte que les modèles ne sont plus cohérents et, par conséquent, qu'une adaptation des diagnostiqueurs est nécessaire.
L'identification des diagnostiqueurs à adapter utilise une heuristique basée sur la mesure de confiance d'un diagnostiqueur envers sa prédiction : parmi les diagnostiqueurs impliqués dans une contrainte non satisfaite, celui qui a la confiance maximale fournit le diagnostic résultat. Ceux qui ne le respectent pas doivent s'adapter en conséquence.

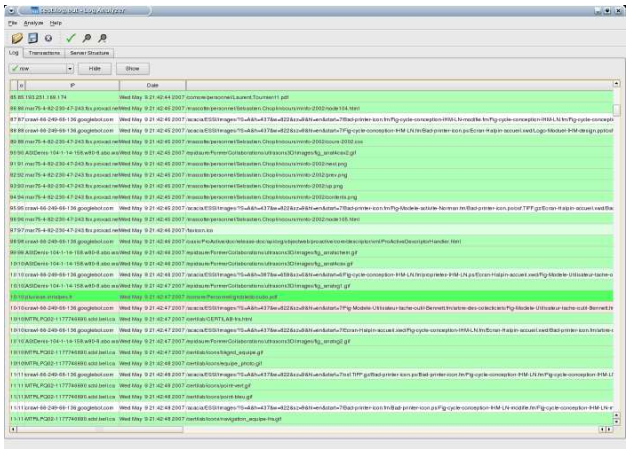

FIG. 2 – Interface de *LogAnalyzer* visualisant le résultat du diagnostic d'un log : la couleur affectée à chaque ligne reflète le diagnostic (vert : normal, rouge : anormal) ; l'intensité exprime la confiance associée au diagnostic.

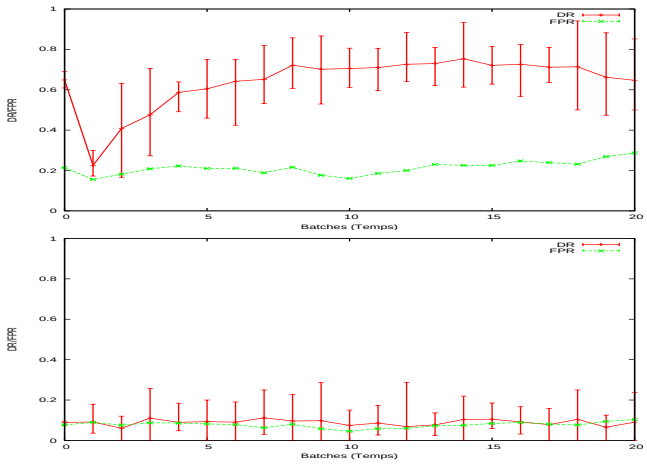

FIG. 3 – Évolution du taux de détection (DR) et du taux de faux positif (FPR) dans le temps. En haut : avec adaptation, en bas : sans adaptation.

2 *LogAnalyzer*

La méthode de détection d'intrusion utilisant un ensemble de diagnostiqueurs a été implémentée sous la forme d'une API (qui peut être réutilisée pour différentes applications) et d'un logiciel d'analyse d'un log d'alarmes structurées : *LogAnalyzer*. La version actuelle est écrite en langage C++ (avec interface en Qt). Elle est diffusée sur le site suivant : `http ://www.irisa.fr/dream/LogAnalyzer/`.
Le logiciel propose trois modes d'analyse :

- un mode exploratoire pour visualiser, structurer (par tris et filtrages) et annoter interactivement les alarmes. Ce mode permet également d'évaluer visuellement les diagnostiqueurs et leurs combinaisons sur de petits jeux de données,
- un mode *hors ligne* qui permet d'évaluer les performances d'un système de détection à partir de grands jeux de données (plusieurs millions d'alarmes),
- un mode *en ligne* prenant en entrée un flux de donnés.

3 Démonstration : détection d'intrusions Web

La détection d'intrusions Web consiste à identifier les requêtes HTTP visant un serveur Web protégé. L'utilisation d'un détecteur adaptatif est fortement motivé par le changement permanent de la "distribution" du flux [3] (*Concept drift*) : les intrusions, le contenu du serveur Web et le comportement des internautes évoluent.
LogAnalyzer a été utilisé comme cadre de développement d'un diagnostiqueur adaptatif de détection d'intrusions capable de conserver de bonnes performances dans le temps tout en limitant les interventions manuelles fastidieuses [2]. Pour cela, nous avons développé plusieurs diagnostiqueurs d'intrusions locaux utilisant différents modèles de requête HTTP permettant de discriminer les intrusions des requêtes normales. Ils utilisent plusieurs descripteurs, comme :

- la distribution de caractères dans l'URL,
- la distribution de tokens (*e.g.* '`htaccess`', '`passwd`'),
- le nombre de requêtes effectuées par un utilisateur.

Nous faisons l'hypothèse que les diagnostiqueurs fournissent des diagnostics identiques pour une même requête d'un utilisateur : soit normale, soit anormale. Muni des contraintes d'intégrité correspondant à cette hypothèse, le système peut découvrir de nouvelles intrusions. Il est également possible de raffiner les modèles d'intrusion en utilisant à la fois les requêtes correspondant à des intrusions et les requêtes normales émises par les autres utilisateurs.
La Figure 3 montre l'évolution de la qualité du diagnostic sur une séquence de batchs de 1000 requêtes chacun. Sans adaptation dynamique des diagnostiqueurs, les intrusions, initialement inconnues, qui ont été introduites artificiellement dans le log ne sont pas détectés. Avec adaptation, ces nouvelles intrusions sont détectées, intégrées dans les modèles et reconnues par la suite. Le taux de détection est alors amélioré au fur et à mesure de l'analyse. Les figures représentent un comportement moyen du système sur 10 distributions d'intrusions différentes, insérées dans un même log de requêtes HTTP. La démonstration présentera les éléments précédents et des exécutions sur des logs provenant des serveurs web de l'IRISA.

Références

[1] T. Guyet, R. Quiniou, W. Wang, and M.-O. Cordier. Self-adaptive web intrusion detection system. Research Report RR-6989, INRIA, 2009.

[2] T. Guyet, W. Wang, R. Quiniou, and M.-O. Cordier. Diagnostic multi-sources adaptatif. application à la détection d'intrusion dans des serveurs web. In *EGC'2009*, pages 325–336, 2009.

[3] F. Maggi, W. K. Robertson, C. Krügel, and G. Vigna. Protecting a moving target : Addressing web application concept drift. In *RAID'2009*, pages 21–40, 2009.