REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

N° d'ordre : .....

UNIVERSITE DE M'SILA

FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE

Département d'Informatique

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Réseaux

Par: LOUNIS Ouarda

SUJET

Implémentation d'un scanner de vulnérabilité

Soutenu publiquement le : / /2013 devant le jury composé de :

.....	Université de M'sila	Président
SAOUDI LALIA	Université de M'sila	Rapporteur
.....	Université de M'sila	Examinateur
.....	Université de M'sila	Examinateur

Promotion : 2012 /2013

TABLE DES MATIERES

INTRODUCTION GENERALE

Contexte et problématique :

La sécurité des applications Web est un problème désormais récurrent. Le nombre de vulnérabilités recensées dans ce type d'applications s'accroît constamment, tel que décrit notamment dans le document « OWASP The Ten Most Critical Web Application Security Risks » **[18]**. Nous pouvons l'expliquer par plusieurs raisons : la complexité sans cesse croissante des technologies du Web, les délais sans cesse plus courts de mise sur le marché de logiciels, les compétences parfois limitées et le manque de culture en sécurité des développeurs. En conséquence, un bon nombre de ces applications contient de multiples vulnérabilités qui peuvent être exploitées par des pirates informatiques. Ces attaques peuvent leur permettre, par exemple, d'obtenir des données confidentielles (numéros de cartes de crédit, mots de passe, etc.) qui sont manipulées par l'application, voire même de modifier ou détruire certaines de ces données. La complexité des technologies utilisées aujourd'hui pour réaliser les applications Web (Java, JavaScript, PHP, Ruby, J2E, etc.) fait qu'il est particulièrement difficile 1) d'empêcher l'introduction de vulnérabilité dans ces applications et 2) d'estimer ou de prévoir leur présence. De plus, la sécurisation des réseaux ainsi que l'installation de pare-feu ne fournit pas de protection satisfaisante contre les attaques Web car ces applications sont publiques et accessibles à tous. Il est donc nécessaire d'auditer régulièrement les applications Web pour vérifier la présence de vulnérabilités exploitables et ceci peut être réalisé notamment par des scanners de vulnérabilités Web.

Méthode :

Nous étudions dans notre projet, deux approches distinctes, l'approche par reconnaissance de messages d'erreur dans les pages de réponse et l'approche par étude de similarité des pages renvoyées par le serveur. Ces deux approche sont utilisées par la plus part des scanners Web afin de détecter les vulnérabilités présentes dans les applications Web. Nous nous focalisons sur la deuxième approche.

L'algorithme implémenté est une version modifiée et améliorée de l'approche de Rim Akrout **[1]**. Elle consiste à envoyer des requêtes HTTP au serveur Web. Ces requêtes sont structurées en trois classes : classe contenant des requêtes syntaxiquement valides, classe contenant des requêtes syntaxiquement invalides et classe contenant des requêtes aléatoires. Les réponses de chaque requêtes seront enregistrées et comparées à une quatrièmes classe qui

ne contient que les pages de références afin d'étudier la similarité entre elles en utilisant l'algorithme de **Levenshtein [17]**, et suivant des tests, on détermine quelles sont les requêtes qui ont bien permet d'exploiter la faille d'injection SQL.

Les performances de l'approche implémentée, seront discutées suivant deux paramètres : le taux de détection et le faux positif. Des graphes sont dessinés afin de bien exprimer les résultats des expérimentations réalisées.

Les requêtes utilisées ont été extraites du projet OWASP (SQL Cheat Sheet) **[18]**.

Objectifs :

La réalisation d'une approche de détection de l'attaque d'injection SQL dans les applications Web en se basant sur l'envoie des requêtes HTTP et l'analyse des réponses de ces dernières. Une bonne approche offrira un grand taux de détection (le pourcentage des pages vulnérables détectées comme vulnérables) et un faible taux de faux positifs (le pourcentage de fausse détection, une page sain détectée comme vulnérable).

Travaux antérieurs :

1. Utilisation de l'approche par reconnaissance de messages d'erreur

W3af **[20]** a été créé par Andres Riancho en 2006. Il est considéré comme l'un des scanners les plus performants, il est écrit en Python. Son architecture modulaire permet aux utilisateurs d'importer et de modifier facilement les différents modules qui le composent. W3af envoie trois requêtes HTTP pour tester la présence d'une vulnérabilité dans une page, les trois réponses associées à ces requêtes sont ensuite analysées. Si elles contiennent des messages d'erreur SQL, W3af informe l'utilisateur que l'application est vulnérable à une injection SQL. Son défaut est aucun mécanisme supplémentaire n'est implémenté pour vérifier si la vulnérabilité existe réellement ou pas, c'est-à-dire si elle est réellement exploitable.

Wapiti **[21]** est un autre exemple qui suit le même principe. Cet outil développé en Python, est capable de détecter des injections SQL, des injections XSS, des mauvaises manipulations de fichiers, des injections LDAP et des exécutions de commandes du système d'exploitation à partir d'une URL. Pour identifier des injections SQL, il envoie les deux requêtes. Une vulnérabilité est déclarée présente si un message d'erreur est identifié dans les réponses

produites. L'efficacité de cette approche est liée à la complétude de la base de connaissance regroupant les messages d'erreurs.

2. **Utilisation de l'approche par étude de similarité des pages de réponses**

Skipfish **[12]** est un outil développé par Google afin de détecter des vulnérabilités sur des serveurs Web. Il procède en deux étapes. Dans une première étape, il parcourt le site et collecte toutes les pages qui lui semblent stables. Les autres sont ignorées. Pour détecter si une page est stable, Skipfish envoie 15 requêtes et plusieurs tests sont réalisés sur la page. Le principal défaut de ce scanner est que la distance utilisée pour l'étude de similarité considère la fréquence des mots sans tenir compte de l'ordre des mots dans un texte.

Structure du mémoire:

Afin de répondre à cet objectif, ce mémoire est structuré de la façon suivante :

Le premier chapitre étudie le contexte de nos travaux. Tout d'abord, il donne un aperçu des technologies Web. Puis, une figure est présentée pour montrer la raison par laquelle les applications Web sont vulnérables. Ensuite, il présente un panorama des vulnérabilités et attaques Web les plus répandues aujourd'hui, en présentant quelques exemples pour chacune d'entre elles. Enfin, deux moyens de sécurité (IDS, Pare-feu) sont discutés.

Le deuxième chapitre, sera consacré à la présentation de l'attaque d'injection SQL en détail qui est l'objet de ce travail. Nous allons expliquer brièvement d'autre faille d'injection. Puis, nous exposerons les différents types de test de sécurité des applications Web. Ensuite, nous parlerons de deux outils existants en mettant en claire leurs limites.

Le troisième chapitre propose une étude détaillée des différentes techniques utilisées par les scanners de vulnérabilités Web existants, ainsi qu'une analyse critique de ces techniques. Nous proposons également notre contribution, qui porte sur la proposition d'une nouvelle approche pour la détection de l'injection SQL dans les applications Web en détaillant les différents types de requêtes utilisés ainsi que la distance textuelle employée. Donnant par la suite deux schémas : le premier donne une vue de haut niveau de l'analyse effectuée et l'autre détaille le fonctionnement de notre approche.

La plateforme d'évaluation, qui constitue le corps de notre travail fait l'objet du quatrième chapitre. Nous allons présenter les résultats obtenus lors de l'exécution de l'approche implémentée sur deux applications Web différentes. Puis, on va montrer ces résultats sous forme des tables et des graphes muni d'une analyse et des explications.

Une conclusion générale vient couronner ce travail, reprenant les points forts et ceux manquants et donne les jalons des travaux de recherche futurs dans ce domaine.

CONCLUSION GENERALE

De nos jours, les applications Web sont à la fois beaucoup plus répandues et beaucoup plus complexes que celles des années 2000. Le développement du Web dynamique et la richesse fonctionnelle qu'offrent les nouvelles technologies du Web permettent de répondre à un grand nombre de besoins. Néanmoins, cette richesse fonctionnelle s'accompagne d'une complexité grandissante, et cette progression va de pair avec une multiplication de nombre de vulnérabilités informatiques publiées, offrant une surface d'attaque conséquente. De ce fait, aucun serveur Web n'est sûr à 100% et les applications Web sont devenues de plus en plus vulnérables et exposées à des attaques malveillantes pouvant porter atteinte à des propriétés essentielles telles que la confidentialité, l'intégrité ou la disponibilité des informations. Pour faire face à ces menaces, il est primordial de développer des techniques efficaces permettant d'aider les développeurs et les administrateurs des applications Web à identifier les vulnérabilités résiduelles dans ces applications et aussi à évaluer l'efficacité de ces outils qui sont conçus pour faire face à des attaques visant à exploiter ces vulnérabilités. Les travaux effectués dans le cadre de ce mémoire visent à atteindre cet objectif.

Nous avons conçu et développé une nouvelle méthode pour la détection de vulnérabilités dans des applications Web, utilisant des tests afin de classifier les pages vulnérables. La mise en œuvre de cette méthode, s'est concrétisée par le développement d'un nouveau scanner de vulnérabilités qui présente la caractéristique suivante : il permet de détecter de façon précise la présence de vulnérabilités dans une application Web dans la mesure où chaque vulnérabilité est réellement exploitée et la requête permettant cette exploitation est fournie, ce qui permet de vérifier si l'attaque a réellement réussie ou s'il s'agit d'un faux positif.

Les résultats que nous avons obtenus sont encourageants. Ils montrent en particulier que l'approche proposée peut effectivement contribuer à l'amélioration de l'efficacité des scanners Web existants pour la vulnérabilité que nous avons considérée, en offrant la possibilité d'automatiser davantage les campagnes d'évaluation.

Perspectives :

Nous pouvons envisager différentes perspectives à ces travaux de recherche, qui concernent à la fois notre approche de détection de vulnérabilités et notre plateforme d'évaluation.

Tout d'abord, en ce qui concerne l'approche proposée pour la détection de vulnérabilités, des optimisations peuvent s'avérer nécessaires comme : la génération de scénarios d'attaques basée sur l'élaboration du graphe de navigation d'un site Web, l'enrichissement de requêtes de façon à être capable de générer la plus grande variété possible d'attaques pour les injections SQL. Dans la même perspective, il est intéressant d'étudier les autres types de vulnérabilités tels que : XPath, OS Commanding, File Include et LDAP.

En ce qui concerne la plateforme d'évaluation proposée, les tests qui ont pu être réalisés à ce jour l'ont été uniquement sur les applications Web qu'on a développé. Il nous semble important de pouvoir généraliser le teste de cette plateforme sur d'autres applications Web disponibles en source libre.

Enfin, et cette perspective est à plus long terme, il nous semble important d'étudier les vulnérabilités pour lesquelles notre scanner n'est aujourd'hui pas adapté, comme les vulnérabilités de type XSS par exemple. Ce type de vulnérabilités est particulier puisqu'il vise non pas l'état du serveur Web, mais le navigateur du client lui-même (au travers d'une vulnérabilité de l'application Web toutefois). Le diagnostic de la réussite ou non de l'exploitation de la vulnérabilité doit se baser sur l'interprétation par un navigateur de la réponse renvoyée par le serveur. Donc, il est nécessaire d'explorer d'autres solutions nous permettant de tester efficacement ces vulnérabilités sur des applications afin de pouvoir à leur tour les inclure dans notre plateforme d'évaluation. Une piste envisagée pourrait être l'intégration d'un navigateur dans cette plateforme.

BIBLIOGRAPHIE ET CITATIONS

[1]R. Akrout, Analyse de Vulnérabilité et Evaluation de Système de Détection d'Intrusion pour les Applications Web, Doctorat, université de Toulouse, 2012.

[2]Martin Arvidson Markus Carlbark, Intrusion Detection Systems – Technologies, Weaknesses and Trends, 2003.

[3]R.Barnett et B.Rectanus, WAF Virtual Patching Workshop : Securing WebGoat with ModSecurity , Breach Security, 2009.

[4]N. Baudoin , M. Karle : NT Réseaux IDS et IPS, 2003/2004.

[5]W. R. Cheswik, S. M. Bellovin, Firewalls and Internet Security, Addison-Wesley, 1994.

[6]M. Curphey, R. Arawo, and M.V. Foundstone, Web application security assessment tools, IEEE Security & Privacy, 2006, pages 32–41.

[7]J. Dahse, A Static Source Code Analyser For Vulnerabilities in PHP Scripts, NDS Seminar, 2012.

[8]Developpez, http://julien-pauli.developpez.com/tutoriels/securite/ developpement-web-securite, consulter le 28/02/2013.

[9]J. Francisco, Realistic Vulnerability Injections in PHP Web Applications, university of Lisboa, 2011.

[10]E. Gencer, Security Testing of Web Based Applications, Master, Norwegian university of science and technology, 2009.

[11]P. Godefroid, M.Y. Levin, and D. Molnar, Automated whitebox fuzz testing, In Proceedings of the Network and Distributed System Security Symposium, 2008.

[12]Google, http://code.google.com/p/skipfish, consulté le 02/05/2013.

[13]C. Gould, Z. Su, and P. Devanbu. JDBC Checker: A Static Analysis Tool for SQL/JDBC Applications, In Proceedings of the 26th International Conference on Software Engineering (ICSE 04) –Formal Demos, 2004, pages 697–698.

[14]W. G. Halfond and A. Orso. AMNESIA: Analysis and Monitoring for Neutralizing SQL Injection Attacks, In Proceedings of the IEEE and ACM International Conference on Automated Software Engineering, Long Beach USA, Nov 2005.

[15]W.G.J.Halfond, J.Viegas, A.Orso, A Classification of SQL Injection Attacks and Countermeasures, Proc. of the International Symposium on Secure Software Engineering, 2006.

[16]Huang, Yao-Wen and Huang, Shih-Kun and Lin, Tsung-Po and Tsai, Chung- Hung, Web application security assessment by fault injection and behavior monitoring, WWW: Proceedings of the 12th international conference on World Wide Web, 2003, pages: 148-159.

[17]V. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals", Soviet Physics Doklady, 1966, pages 707-710.

[18]OWASP, www.owasp.org, consulté le 05/05/2013.

[19]Research, http ://research. cs. ncl. ac. uk/ cabernet /www .laas. research. ec. org/ maftia/deliverables/full.htm#, consulté le 01/01/2013.

[20]Sectools, http://sectools.org/web-scanners.html, consulté le 20/05/2013

[21]Sourceforge, http://w3af.sourceforge.net, consulté le 22/04/2013.

[22]Sourceforge, http://wapiti.sourceforge.net, consulté le 23/04/2013.

[23]K.Stefan, E. Kirda, C. Kruegel and N. Jovanovic, SecuBat : A Web Vulnerability Scanner, Proceedings of the 15th international conference on World Wide Web (WWW '06), Edinburgh, Scotland, 2006.

[24]Wikipedia, http://en.wikipedia.org/wiki/Levenshtein_distance, consulté le 12/04/2013.

Résumé :

De nos jours, La sécurité des applications Web est une nouvelle tendance qui trouve sa nécessité sur le Web. Le nombre de vulnérabilités recensées dans ce type d'applications s'accroît constamment.

Il est donc nécessaire d'auditer régulièrement les applications Web pour vérifier la présence de vulnérabilités exploitables. Notre objectif est La réalisation d'un scanner efficace pour la détection de l'attaque d'injection SQL dans les applications Web en se basant sur l'envoie des requêtes HTTP et l'analyse des réponses de ces dernières en utilisant l'approche par étude de similarité.

Les résultats que nous avons obtenus sont encourageants. Ils montrent en particulier que l'approche proposée peut effectivement contribuer à l'amélioration de l'efficacité des scanners Web existants pour la vulnérabilité que nous avons considérée.

Mots-clefs : attaque SQLI, scanner Web, application Web, vulnérabilités Web, sécurité.

Abstract:

Nowadays, security of Web applications is a new trend that finds its need on the Web. The number of vulnerabilities identified in this type of applications is constantly increasing.

It is therefore necessary to regularly audit Web applications to verify the presence of exploitable vulnerabilities. Our goal is the realization of an effective scanner for the detection of SQL injection attacks in Web applications based on the send of HTTP request and the analysis of responses using the similarity approach.

The results we obtained are encouraging. They show in particular that the proposed approach can effectively contribute to improving the effectiveness of existing Web scanners for the detection of the vulnerability that we have considered.

Keywords: SQLI attack, Web scanner, Web application, Web vulnerabilities, security.

ملخص:

إن أمن تطبيقات الويب هو الاتجاه الجديد الذي يجد حاجته على شبكة الانترنت في الوقت الحاضر .إن عدد نقاط الضعف التي تم تحديدها في هذا النوع من التطبيقات في تزايد مستمر.

ولذلك فمن الضروري مراجعة تطبيقات الويب بشكل منتظم للتحقق من وجود الثغرات. هدفنا هو إنجاز كاشف فعال للكشف عن ثغرات حقن قواعد البيانات SQLفي تطبيقات الويب وهذا من خلال ارسال طلبات HTTP وتحليل النتائج المتحصلة عليها وذلك عن طريق دراسة التشابه بين هذه الأخيرة.

النتائج التي حصلنا عليها مشجعة . وتظهر على وجه الخصوص أن الطريقة المقترحة يمكن أن تساهم بشكل فعال في تحسين فعالية كواشف نقاط الضعف في تطبيقات الويب.

الكلمات المفتاحية : حقن قواعد البيانات,كاشف الثغرات, تطبيقات الويب, ثغرات الويب, الأمن.