Test de vulnérabilité Web à base de patterns et de modèles

Alexandre Vernotte[1], Bruno Legeard[1,2], and Fabien Peureux[1,2]

[1] Institut FEMTO-ST, UMR CNRS 6174 - Route de Gray, 25030 Besançon, France
{avernott, blegeard, fpeureux}@femto-st.fr
[2] Smartesting R&D Center - 2G, Avenue des Montboucons, 25000 Besançon, France,
{legeard, peureux}@smartesting.com

Résumé Cette présentation introduit une approche de génération de tests de vulnérabilité pour les applications Web à partir de modèles et guidée par des patterns de test. Cette approche mixte, baptisée PMVT (Pattern-driven and Model-based Vulnerability Testing), vise à tirer parti des bénéfices et diminuer les faiblesses des techniques de test à base de modèles, généralement utilisées pour le test fonctionnel, et des techniques de test de vulnérabilité, principalement manuelles ou assistées par des scanners de vulnérabilité. Finalement, une synthèse des résultats expérimentaux obtenus et les travaux futurs sont présentés. Ces travaux sont supportés par le projet FSN DAST (http://dast.deptinfo-st.univ-fcomte.fr).

Mots-clés: Test de vulnérabilité, pattern de test de sécurité, test à partir de modèles (MBT), vulnérabilité d'applications Web, Cross-Site Scripting (XSS), injection SQL (SQLI).

1 Contexte et motivations

L'état de l'art actuel sur la sécurité et les différents rapports de sécurité, comme l'OWASP Top Ten 2013 [1], montrent que les applications Web sont aujourd'hui la cible principale des cyber-attaques. Le test de vulnérabilité est une activité de plus en plus pratiquée pour répondre au besoin croissant de sécurité des applications Web. On distingue deux techniques principales : le test de pénétration et les scanners de vulnérabilité. La première approche, généralement manuelle, nécessite un coût humain important et n'est pas exhaustive. La seconde approche est certes peu coûteuse car automatisée mais reste imprécise en détectant un taux important de faux positifs et de faux négatifs [2]. Ce papier propose une approche alternative, baptisée PMVT pour Pattern-driven and Model-based Vulnerability Testing, qui vise à agréger les bénéfices des deux approches historiques tout en maîtrisant leurs faiblesses. Les contributions au test de sécurité se situent :

- dans la capture des comportements applicatifs par un modèle abstrait UML/OCL permettant une couverture de test plus complète de l'application sous test ;
- dans la création d'un langage de script dédié pour minimiser l'effort de modélisation ;
- dans l'extension d'un langage de pattern de test qui permet de guider le générateur de tests à travers le modèle pour couvrir les vulnérabilités à cibler ;
- dans l'automatisation de la génération des tests à partir du modèle et des patterns de test, de l'exécution des test générés, et finalement de la définition du verdict.

2 Principes de l'approche PMVT

L'approche PMVT est une dérivation du Model-Based Testing (MBT) classique pour permettre la génération de tests de vulnérabilité. Cette approche, illustrées en figure 1, est composée des quatre activités principales (une description plus détaillée de chacune des activités est disponible dans [3]). La première activité concerne la définition de *Test Purposes* (①) qui formalisent les objectifs de test de vulnérabilité. Une *Modélisation* (②) de l'application Web en UML/OCL permet ensuite de capturer ses aspects structurels et comportementaux. A partir des artefacts définis lors des deux précédentes activités, on réalise alors une *Génération* automatique de cas de test abstraits (③). Finalement, la phase de *Concrétisation* et d'*Exécution* (④) permet de traduire les cas de test abstraits générés en scripts exécutables, d'exécuter ces scripts sur l'application Web, et d'observer les réponses et de les comparer aux résultats attendus pour assigner un verdict.

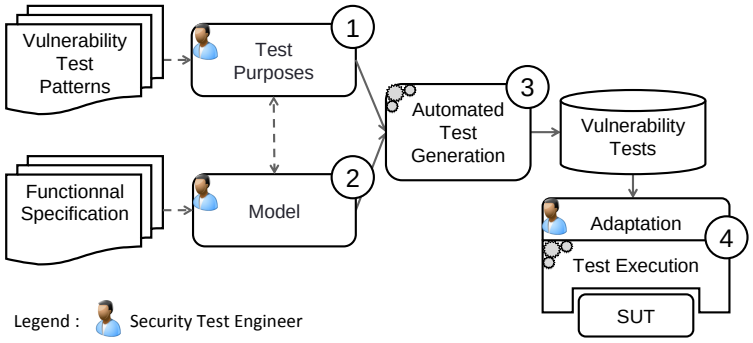

Figure 1. Description de l'approche PMVT

3 Expérimentations et résultats

L'approche PMVT a notamment été expérimentée sur une application de e-learning *Stud-E* (15000 utilisateurs en France) dans l'objectif spécifique de découvrir des vulnérabilités de type multi-step XSS (dans ce type d'injection, les pages d'observation ne se situent pas immédiatement après la page dans laquelle l'injection XSS est réalisée, ce qui nécessite de naviguer dans l'application entre les phases d'injection et d'observation). La mise en œuvre complète de l'approche PMVT, par un ingénieur expert de cette approche, a duré environ douze heures et a donné lieu à la production d'environ 1500 tests et la découverte de 2 failles. Comparativement, nous avons mené une campagne de test de pénétration (avec proxy intrusif) et une campagne d'exécution de scanner de vulnérabilité sur cette même application. D'une part, dix-neuf heures de test de pénétration ont ainsi été nécessaires pour atteindre une couverture de test comparable et détecter les vulnérabilités de type multi-step XSS. D'autre part, l'utilisation de différents scanners (IBM AppScan, NTOSpider, w3af, skipfish, et arachni) n'a pas permis de détecter les 2 failles. Ceci s'explique par la nécessité de connaître non seulement le point d'injection mais aussi le point d'observation.

4 Conclusion et travaux futurs

Cet article a introduit PMVT, une approche pour l'automatisation du test de vulnérabilité Web à partir de modèles et guidée par des patterns de test de vulnérabilité. Les résultats expérimentaux et les études comparatives avec des techniques existantes (test de pénétration et exécution de scanner) ont montré la pertinence de cette approche originale. Ces expérimentations ont également mis en évidence les faiblesses de l'approche PMVT. Elle requiert effectivement un effort encore important de modélisation et de développement (concrétisation des tests) en dépit du langage de description qui permet d'atteindre un premier niveau d'accélération de la phase de modélisation. Pour réduire ces efforts, nous poursuivons en outre plusieurs directions de recherche : d'une part, inférer le modèle UML/OCL (ou une partie du modèle) par utilisation de techniques de Web crawling, et d'autre part de compléter le résultat du crawler avec des traces utilisateurs afin d'obtenir une description suffisante de l'application et conserver un lien entre le modèle UML/OCL et l'application réelle en vue de simplifier la phase de concrétisation. Finalement, nous projetons d'augmenter la couverture des vulnérabilités (notamment les attaques Cross-Site Request Forgery).

Références

1. Wichers, D. : Owasp top 10. `https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project` (October 2013) Last visited : February 2014.
2. Doupé, A., Cova, M., Vigna, G. : Why Johnny can't pentest : an analysis of black-box web vulnerability scanners. In : Proc. of the 7^{th} Int. Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA'10), Bonn, Germany, Springer (July 2010) 111–131
3. Lebeau, F., Legeard, B., Peureux, F., Vernotte, A. : Model-Based Vulnerability Testing for Web Applications. In : Proc. of the 4^{th} Int. Workshop on Security Testing (SECTEST'13), Luxembourg, IEEE CS Press (March 2013) 445–452