Automatização de Teste de Segurança Móvel com MobiSec

Mariano Florencio Mendonça[1], Layse Santos Souza[1], Isadora Lima do Nascimento[1], Fabio Gomes Rocha[1]

[1]Universidade Tiradentes (Unit)
Aracaju - SE - Brasil

{marianofmendonca, santoslay3, isadora.mlima21, gomesrocha}@gmail.com

Abstract. *Evaluating the level of security in Android applications, in an environment that there is a significant amount of data transferred in real time, is not an easy task, because the lack of ability to quickly run the security tests is immense. Therefore, performing the automation of safety tests is an extremely important activity since it is necessary to provide greater safety quality to both the final product and the customer. In this paper, the MobiSec tool is presented that performs the automation of software tests in the Android's operating system's applications through test plans.*

Resumo. *Avaliar o nível de segurança de aplicativos Android, em um ambiente que há uma quantidade significativa de dados sendo transferidos em tempo real, não é uma tarefa fácil, pois a falta de habilidade em executar rapidamente os testes de segurança é imensa. Diante disso, realizar a automatização de testes de segurança é uma atividade de extrema importância, pois é necessário proporcionar uma qualidade de segurança maior tanto ao produto final quanto ao cliente. Neste artigo, é apresentada a ferramenta MobiSec que realiza a automatização de testes de software nos aplicativos do sistema operacional Android através de planos de teste.*

1. Introdução

O mercado de dispositivos móveis vem crescendo gradativamente nos últimos anos para atender a demanda crescente dos aplicativos fornecidos pelas empresas, logo, a segurança desses aplicativos tornou-se importante para garantir a privacidade dos dados processados, e.g., controle de acesso, integridade, autenticação e criptografia. Malek *et al.* (2012) revela que novos desafios na segurança estão surgindo em razão dos ataques, e.g., vírus e malwares, expondo como principal obstáculo a habilidade de avaliar rapidamente a segurança e a robustez de um aplicativo por meio de testes de segurança.

Gargenta (2011) complementa Malek *et al.* (2012) quando revela que esses desafios podem ser solucionados por meio de testes de segurança. O teste de segurança permite avaliar as vulnerabilidades em aplicativos frente a diferentes tipos de ataques de segurança, como também descobrir as novas vulnerabilidades antes que sejam exploradas por atacantes.

De acordo com Kieseberg, Fruhwirt e Schrittwieser (2017) o teste de segurança em aplicativos não é uma tarefa trivial por ser influenciado mediante fatores que tornam complexa a criação de planos de testes eficientes, i.e., variedade de entradas realizadas por sensores diversos, heterogeneidade de tecnologias, variedade de insumos e aumento da conectividade.

StatCounter (2018) afirma que o Android é o segundo sistema operacional mais usado no mundo. Observando a ampliação das possíveis ameaças nos aplicativos, presumimos que se desenvolvêssemos uma ferramenta com o propósito de realizar a automatização dos testes de segurança dos aplicativos em nuvem, esta ferramenta seria capaz de proporcionar uma qualidade maior tanto ao produto final quanto ao cliente. Tal solução foi denominada MobiSec, e optamos por realizar a automatização em nuvem por esta suportar serviços de testes em escala.

2. Fundamentação Teórica

Para o desenvolvimento da solução, MobiSec, foi necessário entender conceitos como Android, Androguard (2012), Androbugs (2014), Androwarn (2012), Inteligência Artificial, AM-TaaS, plataforma mobile, segurança mobile e teste de regressão. Segundo Darwin (2012), o Android é uma plataforma de tecnologia móvel que fornece a telefones celulares, tablets e outros dispositivos portáteis e móveis (até mesmo a netbooks) poder e portabilidade do sistema operacional Linux, confiabilidade e portabilidade de uma linguagem de API de alto nível padronizada.

As ferramentas Androguard, Androbugs e AndroWarn foram escritas em Python, a primeira tem como objetivo criar softwares, exibindo, modificando e salvando seus aplicativos de forma fácil e estática, sendo bastante útil para a realização da engenharia reversa em aplicativos, e.g., malwares. A segunda tem como objetivo levantar potenciais vulnerabilidades por meio da emulação de operações em um aplicativo com base nas vulnerabilidades mais comuns, assim propõe melhorias de acordo com as melhores práticas de desenvolvimento. E por fim, a terceira tem como objetivo detectar e avisar o cliente sobre possíveis comportamentos maliciosos desenvolvidos por um aplicativo Android.

A Inteligência Artificial é um ramo de pesquisa da Ciência da Computação que desenvolve mecanismos e dispositivos que sejam capazes de simular o raciocínio humano, ou seja, estes podem ser treinados e aprenderem com experiências a cumprir tarefas específicas ao processar grandes quantidades de dados e reconhecer padrões. A AM-TaaS, Automated Mobile Testing as a Service, disponibiliza testes automatizados para aplicativos embasado nos critérios de teste publicados pela App Quality Alliance (AQuA).

Shu e Lee (2007) afirmam que a maioria dos testes realizados se limitam a verificar a especificação ou a conformidade da implementação propondo técnicas da inteligência artificial, como aprendizado supervisionado para identificar padrões e falhas na elaboração do software. Sun *et al.* (2009) complementa afirmando que o SVM, Support Vector Machine, por padrão aprende a melhor decisão a ser tomada na maioria dos casos de teste independentemente do método de aprendizado utilizado. O SVM sugere que seja aplicado um limite para as tarefas que envolvem altas taxas de desequilíbrio do que aplicar qualquer estratégia de reamostragem. Sendo assim, o SVM é uma técnica de aprendizado supervisionado de maior eficácia na análise de textos.

Por conseguinte, foram encontrados trabalhos relacionados com propostas similares ao que foi desenvolvido no MobiSec. Dentre eles, as ferramentas MobSF (2016) e ApkTool (2017). O MobSF é um framework automatizado, “tudo em um”, de testes de penetração para aplicativos (iOS/Android/Windows) capaz de realizar testes de performance, análise estática, análise dinâmica e de Web APIs. Logo, é uma ferramenta

bastante completa, porém, não considera planos de teste como o MobiSec. E a ApkTool realiza engenharia reversa em aplicativos Android permitindo acesso ao código fonte, tornando possível depurá-lo passo a passo, apesar de ser usada como ferramenta de segurança, esse não é seu objetivo. Mesmo sendo uma ótima ferramenta, não faz a automatização dos testes, distinto ao que é proposto pelo MobiSec.

3. Metodologia de Pesquisa

Em conformidade com o ponto de vista científico, este artigo pode ser classificado quanto aos objetivos como pesquisa descritiva e aplicada, visto que o MobiSec foi idealizado para ser utilizado pelo nível acadêmico ou profissional com a finalidade de obter conhecimento sobre as técnicas de segurança e testes de software para garantir qualidade na segurabilidade aos aplicativos desenvolvidos.

Para elaboração e execução do Mobisec foi necessário realizar uma pesquisa bibliográfica sobre Testes de Segurança em Aplicativos Android com o intuito de estabelecer o estado da arte atual e identificar ferramentas utilizadas e formas de aplicação. Com base na pesquisa, o desenvolvimento do MobiSec segue as etapas de definição da ideia, público, recursos, objetivo, finalidade, construção de documento técnico do MobiSec, programação do código-fonte, inclusão de recursos e avaliação do MobiSec.

4. Ferramenta MobiSec

Para o desenvolvimento da ferramenta foi escolhido como linguagem de programação o PHP com a intenção de gerenciar a automatização dos testes. O Sistema de Gerenciamento de Banco de Dados utilizado foi o MariaDB visando armazenar as informações coletadas nas execuções dos testes. Também foram realizados ajustes na sintaxe do Androguard para otimizar os resultados.

O cliente, ao realizar o login na ferramenta, cadastra seu projeto para um teste guiado ou realiza um teste não guiado em aplicativos importando a apk. Isto pode ser feito quando o cliente considera seu projeto suspeito, dessa forma, o MobiSec convoca o Androguard para realizar a descompilação da mesma. Na sequência executa automaticamente o Androbugs que é responsável pela avaliação de vulnerabilidade, e em seguida executa o AndroWarn que detecta e alerta os possíveis comportamentos maliciosos. Por fim, a IA analisa os relatórios do Androbugs e AndroWarn e reportar se o aplicativo pode ter ou não malware.

Para avaliar a capacidade de automatização dos testes realizados pelo MobiSec foram coletados na internet vinte e cinco aplicativos com malware e vinte e cinco sem malware, servindo também para identificar a eficácia da IA. Empregamos o algoritmo SVM para classificar os aplicativos, e como resultado obtivemos que em 86% a classificação foi correta e em 14% incorreta, representada na Tabela 1.

Tabela 1 - Matriz de Confusão

Matriz de Confusão		
Sem Malware	Com Malware	
22	3	Sem Malware

4	21	Com Malware

Fonte: Elaborado pelo autor.

Na matriz de confusão, os dados de avaliação são um número ou a porcentagem de previsões corretas e incorretas, no caso da matriz acima temos a quantidade de análises dos relatórios gerados pelo Androbugs dos aplicativos com e sem malware.

5. Conclusão

O desenvolvimento de ferramentas que proporcionem a automatização da análise de segurança em aplicativos acompanhado da inteligência artificial possibilitará um incremento importante no processo de teste de software. Este trabalho apresentou o MobiSec que contém estas características, i.e., o MobiSec tem como propósito a automatização de teste de aplicativos para garantir a segurança, como também uma qualidade maior tanto ao produto final quanto ao cliente. A ferramenta está disponível no GitHub no seguinte link: https://github.com/gomesrocha/MobiSec. O MobiSec é open source, possibilitando o download, modificações e melhorias pela comunidade. Para trabalhos futuros planejamos desenvolver redes neurais, e outros métodos serão testados como Naive Bayes, Árvore de decisão, e entre outros.

Referências

Androbugs. (2014). Disponível em: <https://github.com/AndroBugs/AndroBugs_Framework/>.

Androguard. (2012). Disponível em: <https://github.com/androguard/androguard/>.

Androwarn. (2012). Disponível em: <https://github.com/maaaaz/androwarn/>.

ApkTool. (2017). Disponível em: <https://ibotpeaches.github.io/Apktool/>.

Darwin, Ian F. (2012). Android CookBook. O'Reilly Media.

Gargenta, M. (2011). Learning Android. Sebastopol: O'Reilly Media.

Kieseberg, Peter; Fruhwirt, Peter; Schrittwieser, Sebastian. (2017). Security testing for mobile applications. ERCIM News, 109.

Malek, Sam; Esfahani, Naeem; Kacem, Thabet; Mahmood, Riyadh; Mirzaei, Nariman; Stavrou, Angelos. (2012). A framework for automated security testing of android applications on the cloud. In SERE-C'12, pp. 35-36.

MobSF. (2016). Disponível em: <https://github.com/MobSF/Mobile-Security-Framework-MobSF/>.

Shu, G.; Lee, D. (2007). Testing security properties of protocol implementations - a machine learning based approach. In Proceedings of IEEE ICDCS.

StatCounter. (2018). Disponível em: <http://gs.statcounter.com/os-market-share#monthly-201801-201901-bar>.

Sun, Aixin; Lim, Peng, Ee; Liu, Ying. (2009). On strategies for imbalanced text classification using SVM: A comparative study. Decision Support Systems. 48, 191-201.