ウェブアプリケーションのロジックに依存する攻撃の自動生成による脆弱性検査に関する研究

2015 年度

高　松　勇　輔

主　論　文　要　旨

報告番号	㊙ 乙 第　　号	氏 名	高松 勇輔

主論文題目 ：

ウェブアプリケーションのロジックに依存する攻撃の自動生成による脆弱性検査に関する研究

（内容の要旨）

ウェブアプリケーションの脆弱性は深刻な不正攻撃につながっており，不正攻撃によってクレジットカード番号などの個人情報が流出した事例などが多く報告されている．さらに，77% 以上のウェブアプリケーションに少なくとも 1 つの脆弱性があると報告されており，開発段階においてウェブアプリケーションの脆弱性を検査する必要がある．実際に 85% の企業がウェブアプリケーションのリリース前に脆弱性の検査を行っている．それにもかかわらずウェブアプリケーションに脆弱性が残っている主な要因は次の 3 点である．1 点目は，開発者が脆弱性や攻撃に関する知識を持たないことである．2 点目は，脆弱性や攻撃についての知識があっても防御手法の回避法について知識がないことである．3 点目は，開発者に脆弱性や攻撃，防御手法，その回避手法などについての十分な知識はあるものの防御手法の実装に不具合が残ることである．

本論文では，ウェブアプリケーションのロジックに依存する攻撃を自動的に実行することで脆弱性を検査する手法を提案する．ウェブアプリケーションのロジックとは，ウェブアプリケーションが持つ機能（例: ログイン機能やメッセージ送信機能）を動作させるために必要な手順や情報のことである．提案機構は，こうしたロジックに関する情報を利用してウェブアプリケーションの機能を動作させることで，ロジックに依存する攻撃を実行する．提案機構と同様にウェブアプリケーションへの攻撃を実行することで脆弱性を検査する既存手法があるものの，ウェブアプリケーションのロジックに依存する攻撃を実行することができない．なぜなら，このような攻撃の実行に必要となるロジックに関する情報を収集できないためである．提案機構はウェブアプリケーションの開発段階における利用を想定することで，開発者からこのロジックに関する情報を獲得する．例えば，ウェブアプリケーションのロジックに依存する Cross-Site Request Forgery (CSRF) という攻撃を実行するためにウェブアプリケーションにログインする必要があり，開発者からログインのロジックに関する情報を獲得する．提案機構は，このロジックに関するいくつかの情報を自動的に収集するためにテストフェーズにおいて開発者が行うウェブアプリケーションの動作確認を監視する．これは，ロジックに関する情報を要求することで開発者にかかる負担を軽減するためである．提案機構がウェブアプリケーションのロジックに依存する攻撃を実行することで，これまで攻撃を用いて自動的に検査できなかった脆弱性が検査できるようになる．提案機構が自動的に検査を行うことで，脆弱性や攻撃，防御手法，その回避手法などの知識がない開発者でもこの脆弱性を検査することができる．さらに，提案機構は実際に攻撃を実行することで防御手法の不具合も検査できる．

本手法の有用性を示すために，既存手法では実行できない CSRF と session fixation，visual clickjacking という攻撃に本手法を適用する．提案機構が実行するこれらの攻撃で脆弱性を検査できることを確認するために，さまざまなサイトで利用されているオープンソースのウェブアプリケーションを対象に脆弱性検査を行った．その結果，提案機構が検査した機能に残る脆弱性については全て検出することができた．以上の結果から，本手法は脆弱性を検査するためにウェブアプリケーションのロジックに依存する攻撃を自動的に実行できることがわかった．

SUMMARY OF Ph.D. DISSERTATION

School	Student Identification Number	SURNAME, First name
Science for Open and Environmental Systems		TAKAMATSU, Yusuke

Title

A Study on Vulnerability Detection of Web Applications by Automatic Generation of Logic-Aware Attacks

Abstract

Web application vulnerabilities have become an attractive target for attackers. The vulnerabilities could allow the attackers to steal personal information (e.g., credit card number) and force users to perform unintended financial transactions (e.g., money transfer). Three security vendors reported that more than 77 percent of web applications had at least one vulnerability. To eliminate the vulnerabilities, developers should check for them in the web applications during the development phase. WhiteHat Security reported that 85 percent of organizations perform security testing of their web applications. Nevertheless, many web applications are vulnerable in the wild. There are three causes that make the web applications vulnerable. First, the developers implement no countermeasure against attacks in the web applications because they do not have knowledge on the attacks and the vulnerabilities. Second, incomplete defenses are implemented because the developers do not have knowledge on evasion techniques of the defenses. Third, the web applications remain vulnerable due to implementation mistakes in the defenses.

This dissertation presents automatic detection of vulnerabilities in web applications by performing logic-aware attacks. The logic here means a procedure to operate functions in the web applications (e.g., login procedure). Our technique performs the logic-aware attacks by operating the functions with the logic. Although existing techniques perform pseudo attacks, they cannot do the logic-aware attacks. This is because the existing techniques cannot obtain information on the logic. Our technique obtains the information on the logic from web application developers because our technique is designed to be used in the development phase. For example, to perform Cross-Site Request Forgery (CSRF) as the logic-aware attack, our technique obtains the logic to log in the target applications. Our system automatically collects some pieces of information on the logic while the developers confirm the target applications work well. This is because it releases the developers from the burden of providing the information on the logic. Our technique automatically detects vulnerabilities which the existing techniques cannot do because it can perform the logic-aware attacks. The developers can detect the vulnerabilities with our technique even if they are not familiar with the attacks, the vulnerabilities, the defenses, and the evasion techniques of the defenses. Our technique also detects the incomplete defenses due to implementation mistakes because it carries out the attacks.

To demonstrate the usefulness of our technique, it has been applied to CSRF, session fixation, and visual clickjacking which the existing techniques cannot perform. Our technique obtains the information on the logic from the developers to perform these attacks. In experiments our technique detected CSRF vulnerabilities, session fixation vulnerabilities, and visual clickjacking vulnerabilities in real-world web applications. Our experimental results demonstrate that our technique can detect 11 CSRF vulnerabilities and 6 session fixation vulnerabilities in 5 real-world web applications, and 26 visual clickjacking vulnerabilities in 4 real-world web applications. These results also show that our technique can perform logic-aware attacks to check for vulnerabilities.