Universidade Federal de Pernambuco

Graduação em Ciência da Computação

Centro de Informática
2012.1

Validação de sanitizadores através de execução simbólica e cross-checking de bibliotecas

Proposta do Trabalho de graduação

Aluno: Mateus Araújo Borges
Orientador: Marcelo d'Amorim

1. Descrição

Uma das fontes mais comuns de vulnerabilidades em aplicações web é a ausência de validação adequada de entradas fornecidas pelos usuários. O projeto "OWASP Top ten" [1] categoriza falhas de cross-site scripting (XSS) [2] como um dos riscos de segurança mais perigosos a ser enfrentados por uma organização. Até mesmo grandes companhias como Google e Facebook já tiveram problemas com XSS no passado [3]. A abordagem padrão para proteger instituições contra estas ameaças é *sanitizar* entradas fornecidas pelos usuários. Muitas bibliotecas em java provém funções de sanitização, como Google Closure Templates [4], OWASP HTML Sanitizer [5], JTidy [6] e outras.

O objetivo deste projeto é analizar bibliotecas de sanitização existentes em busca de vulnerabilidades. Mais especificamente, nós queremos encontrar entradas que revelem vulnerabilidades em funções de sanitização existentes. Nossa contribuição é dupla: (1) nós desejamos encontrar entradas que violam a especificação da função de sanitização, e (2) reportar vulnerabilidades não cobertas pelas bibliotecas existentes.

1.1 Como funciona

Suponha que temos um conjunto de funções de especificação da forma "boolean repoOk(String s)" que retornam "true" se a string *s* é corretamente sanitizada para um critério qualquer. Um exemplo de tal função poderia ser "cheque se a string não contém aspas". Nós assumimos que muitas destas funções já existem em bibliotecas porquê os testes (para estas funções) precisam das funções repOk.

Nós fazemos uma distinção entre dois tipos de rotinas de sanitização: Rotinas *bottom-level*, que realizam uma sanitização específica da entrada; e rotinas *top-level*, que utilizam muitas das rotinas bottom-level para sanitizar a entrada. Tipicamente, o usuário usa uma das rotinas top-level na entrada recebida do cliente.

Para cada biblioteca sobre análise, nós executaremos simbolicamente a sua rotina de sanitização top-level. Nós desejamos checar todas as funções repOk associadas com funções bottom-level, e gerar entradas para as funções top-level. Com tais entradas, nós vamos então fazer um *cross-checking* das bibliotecas. Nós criaremos um conjunto de casos de teste compostos das rotinas de sanitização top-level de uma biblioteca específica, e usaremos todas as funções repOk coletadas/criadas de todas as bibliotecas como asserções. Uma violação de uma asserção durante a execução desta suíte de teste implica na detecção de uma das duas situações descritas: (1) uma entrada que viola a especificação foi encontrada, ou (2) existe uma vulnerabilidade que não é coberta pela biblioteca.

2. Cronograma

Atividades\Meses	Março	Abril	Maio	Junho
Revisão bibliográfica e estudo sobre assuntos relacionados à sanitização				
Revisão bibliográfica e estudo sobre assuntos relacionados à execução simbólica de strings				
Preparação do ambiente de execução simbólica de strings				
Escolha das bibliotecas a serem analisadas/ escrita das funções repOk.				
Realização do cross-checking				
Redação do trabalho de graduação				

3. Referências

1. OWASP Top Ten Project 2010. https://www.owasp.org/index.php/Top_10_2010-Main
2. Cross-site scripting. http://en.wikipedia.org/wiki/Cross-site_scripting
3. A Systematic Analysis of XSS Sanitization in Web Application Frameworks. http://www.cs.berkeley.edu/~prateeks/papers/empirical-webfwks.pdf
4. Google Closure Templates. https://developers.google.com/closure/templates/)
5. Owasp Html Sanitizer. http://code.google.com/p/owasp-java-html-sanitizer/
6. JTidy http://jtidy.sourceforge.net/

Assinaturas:

_____

Mateus Araújo Borges
Orientando

_____

Marcelo Bezerra d'Amorim
Orientador

Recife, 13 de abril de 2012