UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Laboratório de Segurança em Computação

ANÁLISE DO MUTIILIDAE II: ANÁLISE DE VULNERABILIDADES WEB

Florianópolis

2015

SUMÁRIO

1 INTRODUÇÃO 3
2 AVALIAÇÃO AUTOMATIZADA 5
2.1 METODOLOGIA OWASP 5
2.2 FERRAMENTA W3AF 5
3 POTENCIAIS VULNERABILIDADES 7
3.1 COOKIE COM INFORMAÇÕES DE LOGIN 7
3.1.1 Descrição 7
3.1.2 Impacto 7
3.1.3 Remediação 7
3.1.4 Vetores de ataque 7
3.2 COOKIE SEM HTTPONLY 7
3.2.1 Descrição 7
3.2.2 Impacto 7
3.2.3 Remediação 8
3.2.4 Vetores de ataque 8
3.3 SQL INJECTION 8
3.3.1 Descrição 8
3.3.2 Impacto 8
3.3.3 Remediação 8
3.3.4 Vetores de ataque 8
3.4 CROSS SITE REQUEST FORGERY (CSRF) 9
3.4.1 Descrição 9
3.4.2 Impacto 9
3.4.3 Remediação 9
3.4.4 Vetores de ataque 9
3.5 APACHE SERVER VERSION 10
3.5.1 Descrição 10
3.5.2 Impacto 10
3.5.3 Remediação 10
3.5.4 Vetor de ataque 10
3.6 CROSS SITE SCRIPTING (XSS) 10
3.6.1 Descrição 10
3.6.2 Impacto 10
3.6.3 Remediação 11
3.6.4 Vetores de ataque 11
4 CONSIDERAÇÕES FINAIS 13

1 INTRODUÇÃO

Este documento refere-se à execução de testes de penetração no projeto da Bry. Os testes que serão aplicados no alvo (comumente conhecidos como PenTest, ou Penetration Tests) são de extrema importância, pois têm como objetivo avaliar a segurança do mesmo através da simulação de um ataque por uma fonte maliciosa. Durante o teste do sistema e o processo de avaliação, descrito nas próximas sessões, serão detalhadas as possíveis vulnerabilidades encontradas.

2 AVALIAÇÃO AUTOMATIZADA

2.1 METODOLOGIA OWASP

Para realizar os testes, tomaremos como base documentos da OWASP (Open Web Application Security Project). Esse projeto é bastante focado em auxiliar as empresas a desenvolverem, operarem e manterem aplicações cada vez mais seguras e confiáveis, e é altamente utilizado como guia para testes, uma vez que distribui ferramentas e tutoriais detalhados abertamente. Um dos documentos publicados a cada três anos pela OWASP é o Top Ten.

2.2 FERRAMENTA W3AF

Antes de aplicarmos qualquer teste, fizemos um reconhecimento automatizado utilizando a ferramenda Zed Attack Proxy(ZAP). Esta plataforma é um projeto da OWASP criado para facilitar a realização de PenTest, possibilitando que estes testes sejam feitos de maneira automatizada e dinâmica.

Utilizando o ZAP, é possível efetuar uma varredura na infraestrutura da aplicação e procurar falhas comuns, principalmente as relacionadas ao OWASP Top Ten, que, como dito anteriormente, serviu de base para nossos testes. A ferramenta trabalha atuando como proxy, interceptando requisições enviadas pelo navegador para a aplicação, possibilitando a análise de todo o conteúdo acessado, permitindo aplicação de filtros para procurar por scripts mal formados e outros erros na aplicação que podem levar ao comprometimento da mesma.

3 POTENCIAIS VULNERABILIDADES

3.1 COOKIE COM INFORMAÇÕES DE LOGIN

3.1.1 Descrição

Algum cookie possui no seu valor a informação de login do usuário.

3.1.2 Impacto

Um atacante pode estar monitorando a rede, ou essa informação pode ficar no cache do navegador, e ter acesso ao login de um usuário.

3.1.3 Remediação

A informação de login não deve existir no cookie, ou então, cifrar o valor do cookie.

3.1.4 Vetores de ataque

- **Url:** http://127.0.0.1/mutillidae/index.php;

3.2 COOKIE SEM HTTPONLY

3.2.1 Descrição

O atributo *HTTPOnly* não está sendo definido no cookie.

3.2.2 Impacto

Sem a definição do atributo *HTTPOnly*, *client side scripts*, como JavaScript, tem acesso ao cookie, assim, caso um atacante consiga injetar código JavaScript na aplicação, ele poderá ter acesso aos cookies.

3.2.3 Remediação

Devem ser definidos para todos os cookies o atributo *HTTPOnly*

3.2.4 Vetores de ataque

- **Url:** http://127.0.0.1/mutillidae/index.php;

3.3 SQL INJECTION

3.3.1 Descrição

As falhas de injeção SQL surgem com o envio de dados não confiáveis que são interpretados como parte de uma consulta SQL. O atacante informa dados que enganam o interpretador e executam comandos que acessam dados sem a devida autorização.

3.3.2 Impacto

Dependendo do escopo da vulnerabilidade o atacante pode ler informações sensíveis do banco, modificá-las e executar operações administrativas.

3.3.3 Remediação

Todas as queries devem ser codificadas utilizando queries parametrizadas, ou utilizar *stored procedures*, ou por último, realizar um filtro nos dados fornecidos pelos usuários, bloqueado caracteres especiais do SQL.

3.3.4 Vetores de ataque

- **Url:** http://127.0.0.1/mutillidae/includes/pop-up-help-context-generator.php; **Parâmetro:** pagename; **Método HTTP:** GET;

- **Url:** http://127.0.0.1/mutillidae/webservices/rest/ws-user-account.php;

Parâmetro: username; **Método HTTP:** GET;

3.4 CROSS SITE REQUEST FORGERY (CSRF)

3.4.1 Descrição

Este tipo de vulnerabilidade força um usuário final a executar ações indesejadas em uma aplicação web em que ele está atualmente autenticado, através de técnicas, como o envio de um link por e-mail.

3.4.2 Impacto

O atacante pode forçar os usuários de uma aplicação a executarem ações de vontade do atacante. Caso este ataque atinja um usuário administrador (com privilégios), pode comprometer toda a aplicação.

3.4.3 Remediação

Ao receber as requisições a aplicação deve ser capaz de identificar se ela partiu do próprio site. Isso pode ser feito através de um parâmetro *hidden* no formulário com um valor aleatório, que deve ser verificado antes de processar a resposta da requisição

3.4.4 Vetores de ataque

- **Url:** http://127.0.0.1/mutillidae/index.php; **Método HTTP:** GET;
- **Url:** http://127.0.0.1/mutillidae/webservices/soap/ws-hello-world.php; **Método HTTP:** GET;
- **Url:** http://127.0.0.1/mutillidae/webservices/soap/ws-user-account.php; **Método HTTP:** GET;
- **Url:** http://127.0.0.1/mutillidae/webservices/soap/ws-lookup-dns-record.php; **Método HTTP:** GET;
- **Url:** http://127.0.0.1/mutillidae/webservices/rest/ws-user-account.php; **Método HTTP:** GET;

- **Url:** http://127.0.0.1/mutillidae/documentation/Mutillidae-Test-Scripts.txt; **Método HTTP:** POST;

3.5 APACHE SERVER VERSION

3.5.1 Descrição

O servidor Apache está com o módulo de status habilitado.

3.5.2 Impacto

Qualquer usuário pode acessar as informações de configuração do Apache.

3.5.3 Remediação

O módulo *mod_ status* deve ser desativado.

3.5.4 Vetor de ataque

Através da url http://127.0.0.1/server-status.

3.6 CROSS SITE SCRIPTING (XSS)

3.6.1 Descrição

Esta falha ocorre quando a aplicação envia dados não confiáveis para o navegador sem o correto tratamento, sendo esses dados códigos que são executados pelo navegador da vítima.

3.6.2 Impacto

O XSS permite que os atacantes executem *scripts* nos navegadores da vítima, roubem sessões, ou redirecionem o usuário para outros sites.

3.6.3 Remediação

Todos os dados informados pelos usuários e os obtidos em uma fonte de dados devem passar por um filtro. O filtro principal é transformar os dados em entidades do HTML.

3.6.4 Vetores de ataque

- **Url:** http://127.0.0.1/mutillidae/includes/pop-up-help-context-generator.php; **Parâmetro:** pagename; **Método HTTP:** GET;
- **Url:** http://127.0.0.1/mutillidae/webservices/rest/ws-user-account.php; **Parâmetro:** username; **Método HTTP:** GET;

4 CONSIDERAÇÕES FINAIS

Apesar do avanço tecnológico no desenvolvimento de ferramentas que realizam testes de segurança de forma automatizada, elas jamais substituirão por completo a análise manual, pois tais ferramentas podem interpretar os dados equivocadamente, não estando preparadas para as especificidades de cada sistema ou para pertubações externas. Assim, esse relatório é o primeiro passo para que a empresa solicitante comece a mitigar os pontos de falhas reportados neste trabalho, enquanto a análise manual é realizada.