Um Web Crawler para Projeções e Análise de Vulnerabilidades de Segurança e Consistência Estrutural de Páginas Web

Cristian Cleder Machado
E-mail: <cristian@cristian.com.br>.
URI - Universidade Regional Integrada - Câmpus de Frederico Westphalen
Departamento de Engenharias e Ciência da Computação

Daniel Heler Pohlmann
E-mail: <ielheler@hotmail.com>.
URI - Universidade Regional Integrada - Câmpus de Frederico Westphalen
Departamento de Engenharias e Ciência da Computação

Eduardo Germano da Silva
E-mail: <eduardo.germano@inf.ufrgs.br>.
UFRGS - Universidade Federal do Rio Grande do Sul
Instituto de Informática

Luis Augusto Dias Knob
E-mail: <luis.knob@inf.ufrgs.br>.
UFRGS - Universidade Federal do Rio Grande do Sul
Instituto de Informática

Resumo

Este artigo descreve o desenvolvimento de um *Web Crawler* para examinar e realizar testes de segurança e análise de vulnerabilidade em informações, as quais possam conter lacunas que permitam prejudicar a integridade e confiabilidade de páginas web. Com os resultados, foi possível gerar projeções dos problemas, possíveis causas e recomendações, afim de alertar e auxiliar os administradores dos sites sobre os possíveis problemas, indicando um tratamento próximo ao ideal. Para validação da proposta foram executados experimentos em sites reais. Os resultados dos experimentos mostram a eficiência da ferramenta, detectando problemas em todos os sites analisados e indicando possíveis soluções. Além disso, todos os processos foram realizados corretamente sem prejudicar as estruturas e as informações das páginas dos sites, tornando sua utilização de importante para a redução dos ataques existentes, auxiliando os desenvolvedores, administradores e responsáveis dos web sites, que não tiveram conhecimento dessas falhas ou da gravidade que elas podem trazer.
Palavras-chave: web crawler, segurança, vulnerabilidades, ataques, páginas web

Introdução

Com o avanço do uso da internet e o crescente número de pessoas com acesso a sites, torna-se cada vez mais necessário os programadores e responsáveis se preocuparem com a segurança e consistência da estrutura e informações de seus web sites. Porém, em diversos sites, páginas são

DOI: 10.18256/2359-3539/reit-imed.v2n2p3-12

desenvolvidas e disponibilizadas para os usuários sem a implementação de técnicas para prevenção contra hacking, mesmo para técnicas há muito tempo conhecidas (Castillo, 2005). Entre essas técnicas que podem ser prevenidas pelos programadores, estão os *Cross-site Scripting* (XSS), *Remote File Inclusion/Injection* (RFI), *Directory Traversal* (LI) e o mais conhecido, o *SQL Injection*. Essas falhas procuram explorar parâmetros que a página irá utilizar na comunicação, realização de tarefas, conexões com bancos de dados, utilização de *iframes*, filtros de pesquisa de conteúdo, utilização de áreas restritas, formulários, *links*, entre outros (Scambray, Shema, & Sima, 2006).

A Consistência estrutural das páginas é outro ponto importante, não só para segurança, mais pela facilidade de utilização e compreensão de cada página web. Muitos sites podem ter irregularidades de códigos, arquivos e imagens corrompidas em suas páginas, prejudicando a entidade visual e da navegação, causando um desconforto e insegurança para o usuário permanecer no site.

Atualmente, a necessidade de ferramentas, tecnologias e programas que informem os administradores e os responsáveis de seus sites que em determinada página encontra-se um possível problema, auxiliando com uma projeção de como este poderá ser resolvido, pode ser de importante para prevenir e reduzir ações de má índole e problemas futuros. Diante disso, um *Web Crawler* pode ser uma das soluções para buscar sites identificando e analisando possíveis falhas que as páginas podem conter.

Diante disso, esse trabalho apresenta um *Web Crawler* para encontrar vulnerabilidades em sites que estão na Internet, e que utilizem o *Hypertext Transfer Protocol* (HTTP). Para al- cançar isso, foram inseridas funções para identificar e analisar possíveis falhas na consistência estrutural das páginas que podem gerar algum tipo de vulnerabilidade aos sites. Diferente de outros *Web Crawlers*, o *Web Crawler* proposto explora links externos (por exemplo, a inserção de feed RSS, busca de previsão do tempo ou cotações de outros sites, etc.) que estão ligados de alguma maneira nas páginas do site analisado, afim de identificar a consistência e o tempo de processamento para carregamento dos mesmos. Mais além, outro diferencial é a coleta de informações para alertar por e-mails aos responsáveis e administradores dos sites as análises dos resultados e possíveis soluções para os problemas. Para validação da proposta foram realizados experimentos em 30 sites disponíveis atualmente na Internet. Como resultado, a ferramenta identificou em 23 dos 30 sites analisados algum tipo de vulnerabilidade ou problema de consistência estrutural, demonstrando a viabilidade do uso da mesma.

O restante do texto está organizado da seguinte forma: a Seção *Ataques, Consistência e Vulnerabilidades* apresenta os conceitos dos ataques estudados e como prevenir as vulnerabilidades, além de informar a importância da consistência estrutural. A Seção *Web Crawler* apresenta brevemente o *Web Crawler* proposto e suas funcionalidades. Na Seção *Experi- mentos e Resultados* são apresentados e discutidos os resultados iniciais obtidos. A Seção *Trabalhos Relacionados* apresenta uma breve comparação do *Web Crawler* proposto com outros *Web Crawlers*. A Seção *Considerações Finais* conclui o trabalho, mostrando suas principais contribuições e apontando direções para trabalhos futuros.

Ataques, Consistência e Vulnerabilidades

A maioria das técnicas utilizadas para atacar as páginas de sites são as mesmas utilizadas desde tempos passados até os atuais. Isso ocorre, na maioria das vezes, pelo fato de que mui- tos desenvolvedores não ter conhecimento sobre diversos tipos de ataques. Além disso, outro motivo é o fato de que, devido a demanda de projetos, o site é disponibilizado rapidamente para o usuário, sem a busca e a análise de possíveis falhas ou vulnerabilidades.

Existem diversos tipos de ataques, cada qual, com características diversas. O *SQL Injection* permite a alteração de forma maliciosa dos comandos SQL que são enviados ao banco de dados através de *Uniform Resource Locator* (URL). O mesmo pode ser realizado por alguma entrada de informação em alguma página que tenha comunicação com o banco de dados, podendo inserir, alterar, consultar e excluir informações armazenadas (Dwivedi & Lackey, 2008; Bau, Bursztein, Gupta, & Mitchell, 2010).

O ataque *Cross-site Scripting* (XSS) permite inserir *scripts* em uma parte da página sem permissão em uma zona privilegiada para ser executado quando acessado. Como no *SQL Injection* ele é utilizado em formulários, parâmetros ou alguma parte da página que con- tenha algum tipo de troca de informações. Os sucessos desse ataque po-

dem causar dano na consistência estrutural, roubo de informação, comprometer parte da segurança do site e do servidor, realização de *phishing*, inserir vírus nas páginas, entre outros (Doupé, Cova, & Vigna, 2010; Dwivedi & Lackey, 2008).

O *Remote File Inclusion* (RFI) permite, através de comandos na URL, inserir outros arquivos para serem executados pelo servidor e formar um único arquivo. Os resultados dessa ação podem possibilitar roubo de informação e, em alguns casos, danificar toda a estrutura do site e do servidor de hospedagem. Em última análise, o *Directory Traversal* explora uma falha que permite o acesso a arquivos e/ou diretório de um servidor sem verificar a permissão para leitura de arquivos, podendo, em alguns casos, mostrar configurações e falhas de segurança do servidor (Scambray et al., 2006).

Além de ataques, existem diversos problemas referentes a consistência estrutural de um site. Problemas de consistência estrutural se devem ao fato de conter irregularidades de códigos, arquivos e imagens corrompidas, código fonte exposto, *scripts* sem resposta, prejudicando a integridade visual e a navegação, causando certo desconforto e insegurança para o usuário. Para identificar vulnerabilidade em sites é preciso a realização periódica de testes sem com- prometer a estrutura das informações do mesmo e nem prejudicar o servidor de hospedagem. Esses testes variam de acordo com a vulnerabilidade e devem ser realizados sem comprometer ou expor informações sigilosas. Os comandos e instruções devem ser utilizados apenas para identificar a vulnerabilidade do tipo de ataque. Para identificar a competência estrutural deve ser analisando seu código fonte com os padrões da W3C, além de verificar se não contém nem um arquivo com algum problema HTTP ou corrompido (Doupé, Cavedon, Kruegel, & Vigna, 2012).

Web Crawler

A presente seção faz uma abordagem geral da ferramenta desenvolvida, apresentando as funcionalidades e as ferramentas utilizadas para o seu desenvolvimento. O principal objetivo é projetar e desenvolver um *Web Crawler* para coletar páginas de sites da internet e realizar análise de informações sobre a vulnerabilidade da segurança e na consistência estrutural, gerando projeções de correções através dos resultados obtidos.

Entre as diferenças do *Web Crawler* em relação as demais ferramentas existentes está sua capacidade de poder coletar e realizar testes de forma automática em todos os sites existen- tes na Internet. Para isso, o *Web Crawler* recebe como entrada um determinado site e, ao encontrar links externos no site de entrada, cria uma lista de novas varreduras, que é execu- tada em sequência após o término do processo de análise para cada site encontrado através desses links. Além disso, outro diferencial são as projeções dos resultados das análises, que são encaminhados para os possíveis responsáveis de cada site.

O *Web Crawler* também conhecido como *Web Spider*, *Bots Scutter*, *Bot Crawler* e *Automatic Indexer*, é um programa e/ou *script* focado para a *World Wide Web* (WWW). Um *Web Crawler* é muito utilizado para atualizar bases de dados de motores de buscas, através das informações que estão disponíveis nas páginas web publicadas, tais como links, palavras chaves, *tags*, entre outras. Além disso, um *Web Crawler* pode ser usado para manutenção de web sites, para validar código HTML e checar links (Bau et al., 2010; Doupé et al., 2012). Podem também ser usados para obter tipos específicos de informações de páginas, como endereços de correio eletrônico de e-mails e criar cópias de páginas. Atualmente, também existem *Web Crawlers* para web semântica, a qual possui dados representados por um padrão e seguindo alguns princípios, como tornar as informações mais legíveis, e atribuir algum significado ou sentido na palavra informada pelos usuários (Heydon e Najork, 1999; Doupé et al., 2010).

Para desenvolver o *Web Crawler* foi utilizada à linguagem Python, que possui eficientes estruturas de dados de alto-nível e uma simples e efetiva abordagem para programação orientada a objetos, além de suporte a diversas APIs e bibliotecas que auxiliam na programação e no desenvolvimento do *Web Crawler*. Um banco de dados MySQL foi utilizado para armazenar as informações coletadas pela ferramenta e as instruções, comandos, e informações essenciais que são utilizadas durante a execução do *Web Crawler*.

A Figura 1 apresenta o diagrama de execução. O programa começa com a inicialização de todas as bibliotecas necessárias e suas funções. Além disso, é feita a conexão com o banco de dados e a maioria das consultas de informações necessárias para serem utilizadas por cada função. Esse procedimento é realizado para evitar uma sobrecarga no banco de dados ao realizar diversas

consultas. Após isso, o primeiro site é extraído do banco de dados com o objetivo de ser coletadas as páginas e a realização dos testes e análises.

O próximo passo é identificar a existência de e-mails nos sites. Para coletar os e-mails, é utilizada a biblioteca "re" do Python responsável pela utilização de expressões regulares utilizando regex, facilitando à coleta de informações que seguem um padrão e tornado mais rápido a consulta.

Para identificar formulários na página, foi utilizada a mesma técnica empregada para a coleta de e-mails. Quando algum formulário é identificado, é coletado o destino do formulário e o método utilizado para encaminhar os parâmetros do tipo POST ou GET. Para coletar essas informações é utilizada a biblioteca "BeautifulSoup", que possibilita utilizar expressões e métodos simples para navegar, pesquisar e modificar, ou seja, um conjunto de ferramentas para extrair as informações necessárias de um documento.

Após a coleta dos formulários é ajustado o link do destino das informações para manterem um padrão, ou seja, tornar todos os links uma URL completa. Também são coletados os nomes dos campos desse formulário e os campos que tiverem algum valor por padrão. Após isso, todos os campos que não possuem um valor por padrão são inseridos como uma ins- trução específica de cada vulnerabilidade para serem enviadas ao site através de um método estabelecido, verificando se houve sucesso no ataque.

Para realizar os testes de consistência estrutural é utilizada uma função que envia uma requisição para o site <http://validator.w3.org/check?uri=> e manda como parâmetro a URL da página. Caso seja identificado algum tipo de problema na estrutura do site, esta é salva no banco de dados.

A coleta de links é o último passo realizando em cada página. Primeiramente é utilizado o regex para verificar os links que estiverem dentro do conteúdo da página, ignorando o que esta fora das *tags* HTML "<body></body>". Através desse conteúdo, serão identificados e analisados os *links* que estão dentro das *tags* do HTML "href" e "src". Além disso, é coletado o caminho da página encontrada para ser utilizado na hora de ajustar os links que não possuem um caminho absoluto. Os links são ajustados e verificados se sobre a ocorrência de algum problema HTTP ou se o arquivo não esta corrompido. Caso seja um link que leva a um tipo de página que pertence explicitamente ao site, o mesmo é inserido numa fila de URLs no banco de dados para ser analisado posteriormente. Caso a página pertença a outro site, o mesmo é inserido no banco de acessos externos.

No passo seguinte, é verificado se não existe uma nova página naquele site para ser coletada. Caso não exista, é verificado se não existe outro site para analise. Após varrer todas as páginas dos sites, é iniciado um processo de testes dos sites ainda não analisados. Cada site coletado e testado tem seus resultados armazenados em um banco de dados. Esses resultados podem ser usados de diferentes maneiras, tendo como foco principal, contribuir com a redução de acontecimentos atuais sem prejudicar e expor informações sobre cada site. Essas utilizações envolvem pesquisa, estatísticas, documentações, e sistemas para diferentes tipos de empresas.

Com objetivo de enviar esses resultados para os possíveis administradores e responsáveis de cada site verificado, foi desenvolvido um portal que auxilia na análise e indica possíveis soluções para os problemas encontrados. O portal esta dividido em duas partes: a primeira tem como objetivo administrar as informações do *Web Crawler* ; a segunda, vai disponibilizar o acesso às informações e projeções para os usuários e administradores responsáveis do site. O portal requer autenticação para garantir a integridade e confiabilidade das informações de cada página exposta, sem riscos de acessos não autorizados. A Figura 2 apresenta um esboço da tela de informações sobre *SQL Injection*.

Experimentos e Resultados

Os experimentos foram realizados em 30 sites disponíveis atualmente na Internet. Os 30 sites foram retirados do top 500 mais acessados do Brasil, fornecido pelo site Alexa[1]. Para não induzir ataques aos primeiros 30 sites da lista, os 500 sites foram inseridos em um sistema de randomização[2] que escolheu os 30 sites a serem analisados. Além disso, por questões de segurança e sigilo, os domínios e URLs foram ocultadas na apresentação dos resultados. O objetivo dos experimentos foram demonstrar a eficiência da ferramenta na detecção de vulnerabilidades e a precisão na indicação de possíveis soluções.

Referente ao resultado dos experimentos, 23 sites dos 30 analisados apresentaram algum tipo de vulnerabilidade ou problema de consistência

1 http://www.alexa.com/topsites/countries/BR
2 http://www.random.org/lists/

estrutural. Os resultados obtidos destes 23 sites podem ser observados na Figura 3. Como pode ser observado, foram detectadas as vulnerabilidades de *SQL Injection* em 19 sites, de XSS em 5 sites, e de RFI em 3 sites. A surpresa é que, segundo a análise realizada através do site W3C[3]3 , todos os sites possuem algum tipo de problema na consistência estrutural, porém, isso não significa que todas as páginas dos sites contém algum problema de consistência.

No total de sites analisados foram encontradas 591 páginas com problemas. Como apresentado na Figura 4, dos 19 sites que foram identificadas vulnerabilidades de *SQL Injection*, 39 páginas apresentam essa vulnerabilidade, enquanto 25 páginas possuem vulnerabilidade de XSS e 3 páginas estão vulneráveis a RFI. Apesar de todos os sites apresentarem algum pro- blema de consistência estrutural 84 páginas de alguns desses sites estão seguindo os padrões W3C sem nem um problema identificado.

Em relação à vulnerabilidade de *SQL Injection* de 39 páginas que apresentaram esse problema, cerca de 94,87% foram identificadas nas URL que possuem parâmetros para passar informações entre as páginas e o banco de dados, isso corresponde a 37 páginas. Apenas 5,13% dessas vulnerabilidades foram identificadas em formulários HTML, como mostra a Figura 5.

Por outro lado, no ataque de *Cross-site Scripting*, das 25 páginas, 28% foram identificados em formulários HTML utilizando o método POST, e 40% em formulários utilizando o método GET. Além disso, 32% foram identificados através de parâmetros de URLs, como mostra a Figura 6.

Os ataques do tipo *Remote File Inclusion* foram todos identificados através de parâmetros por URL. Não foram identificadas vulnerabilidades através do ataque *Directory Traversal*. Como mostra a Figura 7, entre as 507 páginas que apresentaram algum tipo de erro em sua estrutura, cerca de 4,73% possuem acima de 100 erros. Além de problemas com a consis- tência referente aos padrões W3C foram verificados os arquivos, páginas e documentos que contidos nos *links* dos sites. A maioria dos erros identificados são do tipo HTTP, os quais são apresentados pelos próprios servidores de hospedagem quando ocorrido algum problema ao acessar algum tipo de arquivo. Em 86,96% dos sites, foram identificados *links* com al- gum tipo de problema e somente 13,04% não apresentaram problema algum. É importante ressaltar que erros estruturais nos sites, na maioria dos casos, não afetam em vulnerabilida- des. No entanto, eles podem implicar em problemas parciais ou completos de visualização das informações, trazendo desconforto aos usuários dos mesmos. A Figura 8 mostra esses resultados.

3 http://www.w3.org/

Trabalhos Relacionados

Atualmente existem diversas ferramentas para realizar varreduras em aplicações web com objetivo de procurar vulnerabilidade e/ou consistência estrutural. Entre as diferenças do Web Crawler em relação as demais ferramentas existentes, está sua capacidade de poder coletar e realizar testes de forma automática. A partir de uma simples entrada, o Web Crawler pode varrer em cascata os sites encontrados dentro de um site inicial.

Na Tabela 1 são apresentadas comparações de ferramentas gratuitas com objetivos de identi- ficar vulnerabilidade e/ou consistência estrutural. Essas ferramentas são recomendadas pela Open Web Application Security Project (OWASP)[4], organização sem fins lucrativos com objetivo de melhorar a segurança de softwares em todo mundo. Como pode ser observado na Tabela 1, o *Web Crawler* proposto neste trabalho possui todas as características apresenta- das pelos *Web Crawlers* comparados. Além disso, ele explora links externos (por exemplo, a inserção de feed RSS, busca de previsão do tempo ou cotações de outros sites, etc.) que estão ligados de alguma maneira nas páginas do site analisado, afim de identificar a consistência e o tempo de processamento para carregamento dos mesmos. Mais além, outro diferencial é a coleta de informações para alertar por e-mails aos responsáveis e administradores dos sites as análises dos resultados e possíveis soluções para os problemas.

Considerações Finais

Este artigo apresenta uma solução para identificar e indicar como diminuir o número de ocorrências de tipos de ataques que possam comprometer a segurança e a consistência da informação em sites. O *Web Crawler* coletou páginas de sites que estão na rede Internet que utilizam o protocolo HTTP, e realizando testes de segurança e aná-

4 https://www.owasp.org

lise de vulnerabilidade em informações, as quais podiam conter falhas que permitam prejudicar a integridade e confiabilidade de páginas web. Além de verificar a consistência estrutural que possam comprometer a visualização das informações.

A ferramenta para administrar as informações coletadas pelo *Web Crawler* e gerar projeções além do painel para os usuários acessarem essas projeções foi projetada de forma intuitiva. O objetivo é fazer com que a mesma possa ser utilizada por empresas ou como *plugin* para outras aplicações.

Com os resultados obtidos foi possível gerar projeções para serem encaminhadas para seus responsáveis, alertando sobre os problemas que foram encontrados com informações, dicas e soluções. Essas projeções podem ser de importantes para os desenvolvedores, administrado- res e responsáveis dos sites, que não tiveram conhecimento dessas falhas ou da gravidade que essas falhas podem trazer para o site antes que a ferramenta realizasse os testes, podendo comprometer a segurança do site e dos usuários.

Os resultados obtidos foram satisfatórios para comprovar que suas funções estão realizando os processos corretos sem prejudicar a estrutura e informações das páginas dos sites, tornando sua utilização importante para a redução de ataques existentes. Além disso, essas informações anonimamente podem ser utilizadas para estudos, estatísticas, documentações e normas que contribuem para os sites que estão sendo desenvolvido e os que estão disponíveis se tornarem seguros e consistentes.

Esse *Web Crawler* pode se tornar uma das ferramentas que auxiliam administradores, desenvolvedores e responsáveis de site a se prevenirem e evitarem que seus sites sejam vitimas de ataques e invasões que possam comprometer sua estrutura e informação. E tudo isso sem a necessidade do cliente estar indo atrás desses recursos, pois a ferramenta pode identificar seu site na internet para ser testado. Além de poder oferecer a possibilidade de se criar um sistema de gerenciamento, administração e de informação online para atender esses sites. Como trabalhos futuros pretende-se, melhorar e enriquecer cada função, de forma a aperfei- çoar os seus resultados. Desenvolver novas funções para outros tipos de falhas que podem conter as páginas web. Adquirir maior estudo na linguagem com objetivo de aperfeiçoar fun- ções e poder organizar melhor o código fonte, além disso, utilizar as bibliotecas *Python* para criar *threads* e/ou programação distribuída em seus testes para melhorar a coleta dos sites e dos testes, otimização de recursos e tempo de funcionamento. Além de melhorar o sistema para gerar projeções, com possibilidades de realizar todas as funcionalidades automáticas, podem ser adotadas outras ideias que contribuem para o mesmo propósito. Melhorar as instruções e informações do Banco de Dados para manter sempre atualizado.

Referências

Bau, J., Bursztein, E., Gupta, D. & Mitchell, J. (2010). State of the art: automated black- box web application vulnerability testing. Em *Security and privacy (sp), 2010 ieee symposium on* (pp. 332–345). IEEE.

Castillo, C. (2005). Effective web crawling. Em *Acm sigir forum* (Vol. 39, *1*, pp. 55–56). ACM.

Doupé, A., Cavedon, L., Kruegel, C. & Vigna, G. (2012). Enemy of the state: a state-aware black-box web vulnerability scanner. Em *Presented as part of the 21st usenix security symposium (usenix security 12)* (pp. 523–538).

Doupé, A., Cova, M. & Vigna, G. (2010). Why johnny cant pentest: an analysis of black-box web vulnerability scanners. Em *Detection of intrusions and malware, and vulnerability assessment* (pp. 111–131). Springer.

Dwivedi, H. & Lackey, Z. (2008). *Hacking exposed web 2.0: web 2.0 security secrets and solutions*. McGraw-Hill Publishing.

Grabber. (2016). Grabber web server scanner. Recuperado de http://rgaucher.info/beta/grabber/

Grendel-Scan. (2016). Grendel-scan web server scanner. Recuperado de http://sectools.org/tool/grendel-scan/

Heydon, A. & Najork, M. (1999). Mercator: a scalable, extensible web crawler. *World Wide Web*, *2* (4), 219–229.

Nikto. (2016). Nikto web server scanner. Recuperado de https://cirt.net/Nikto2

Proxy, Z. A. (2016). Zed attack proxy. Recuperado de https://www.owasp.org/index.php/ OWASP_Zed_Attack_Proxy_Project

Scambray, J., Shema, M. & Sima, C. (2006). *Hacking exposed: web applications*. McGraw-Hill New York.

Vega. (2016). Vega web server scanner. Recuperado de https://subgraph.com/vega/ Wapiti. (2016). Wapiti web server scanner. Recuperado de http://wapiti.sourceforge.net/

Wikto. (2016). Wikto web server scanner. Recuperado de http://sectools.org/tool/wikto/

Tabela 1. Comparativo entre Web crawlers

Nome	*Plataformas*	*SQL Injection*	XSS	RFI	LFI	Consistência Estrutural
Web Crawler proposto	Plataformas que supor- tam python 2.7	x	x	x	x	x
(Grabber, 2016)	Plataformas que supor- tam python 2.4	x	x	x		x
(Grendel-Scan, 2016)	Windows, Linux e MacOS	x	x			
(Nikto, 2016)	Unix/Linux	x	x	x	x	
(Vega, 2016)	Windows, Linux e MacOS	x	x	x		
(Wapiti, 2016)	Windows, Unix/Linux e MacOS	x	x			
(Wikto, 2016)	Windows	x	x	x	x	x
(Zed Attack Proxy, 2016)	Windows, Unix/Linux e MacOS	x	x	x		

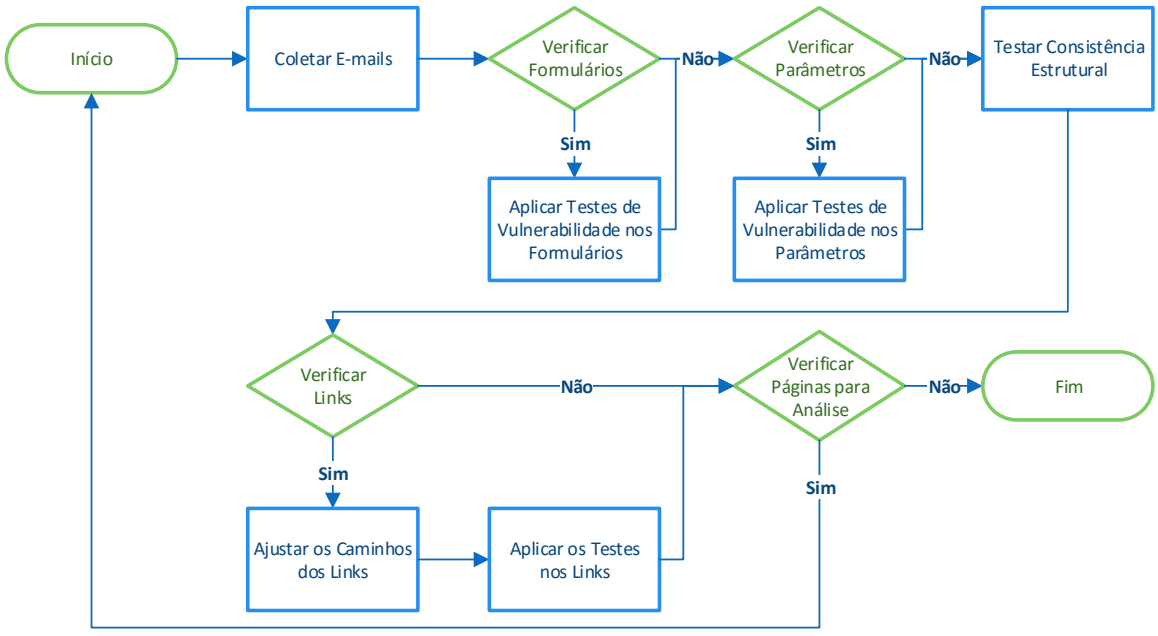

Figura 1. Diagrama de funcionamento do Web Crawler

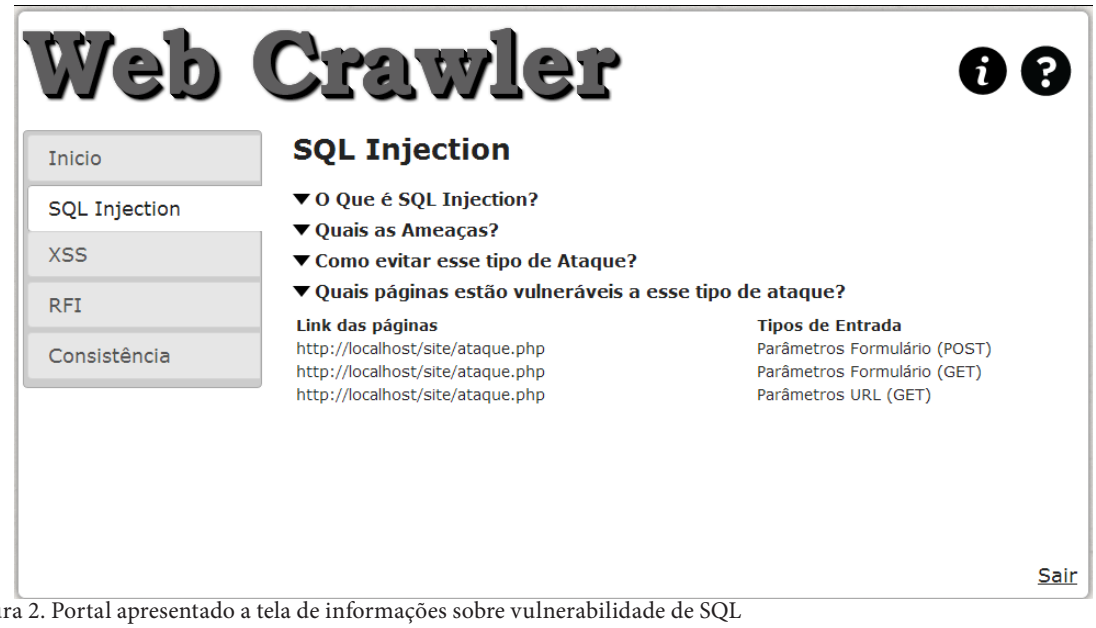

Figura 2. Portal apresentado a tela de informações sobre vulnerabilidade de SQL

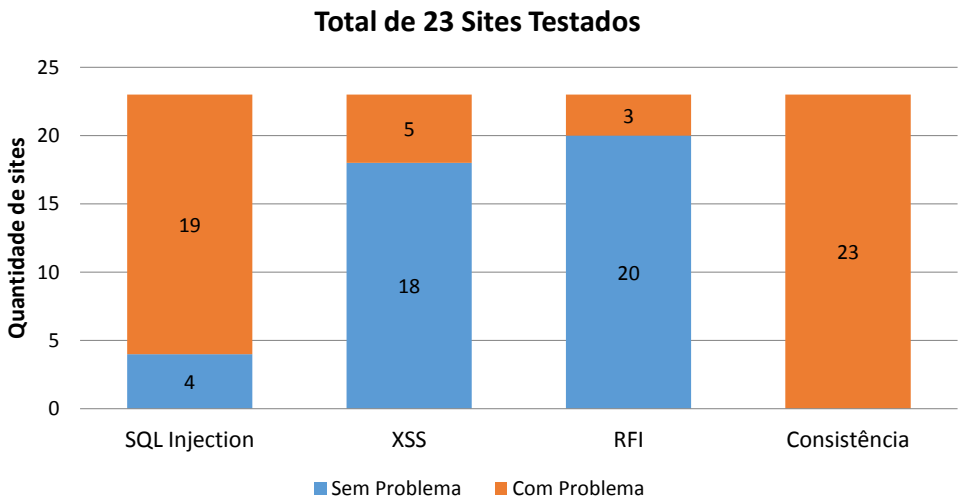

Figura 3. Resultado total dos testes nos 23 sites com problemas

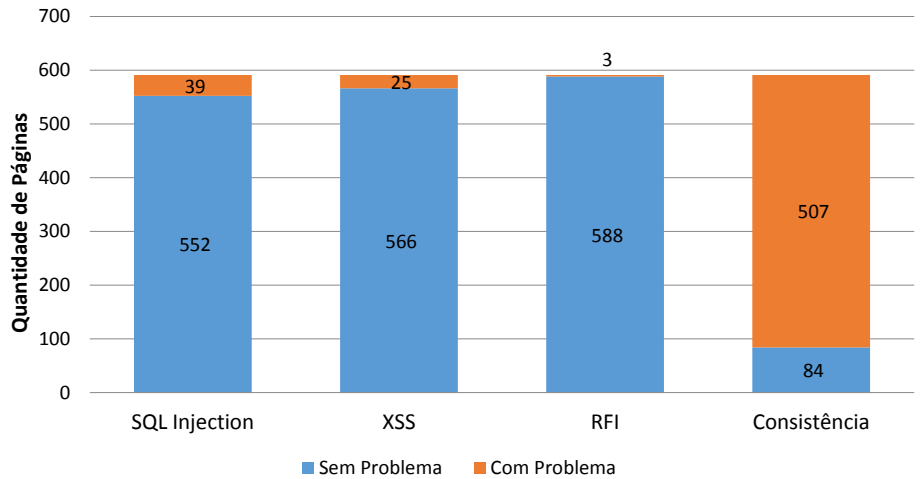

Figura 4. Total de 591 páginas analisadas pelo Web Crawler

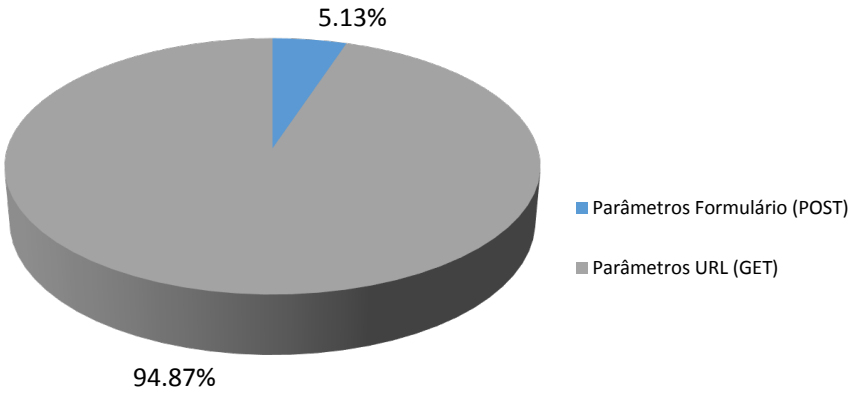

Figura 5. Métodos identificados: Vulnerabilidade de SQL Injection em 39 páginas.

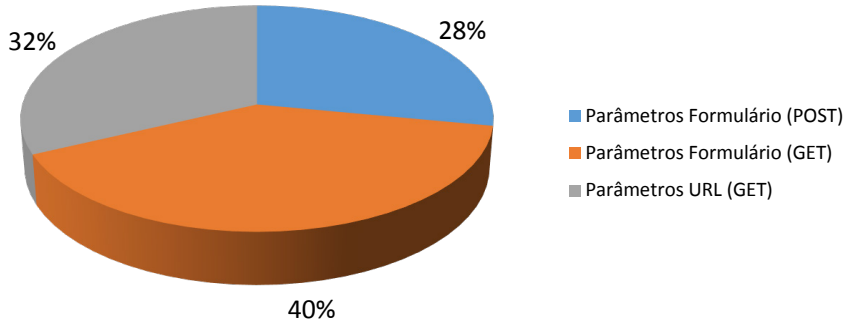

Figura 6. Métodos identificados: Vulnerabilidade de XSS em 25 páginas.

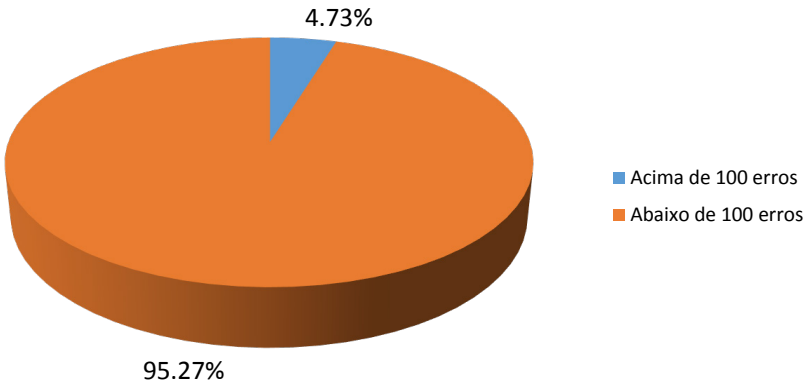

Figura 7. Quantidade de problemas na consistência estrutural em 507 páginas.

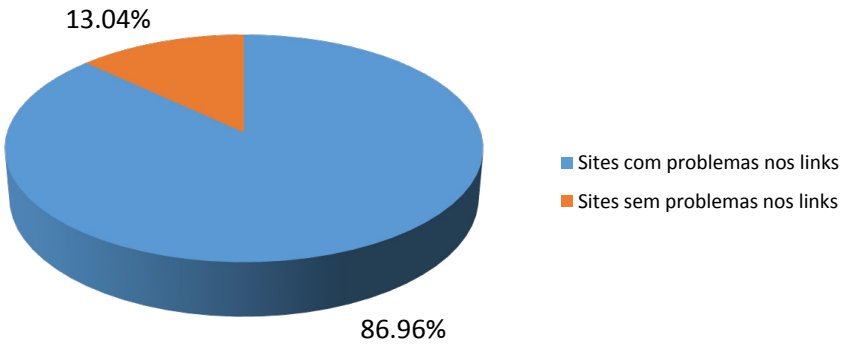

Figura 8. Consistência dos links de 23 sites.

Abstract

This paper describes the development of a Web Crawler to examine and carry out securety tests and vulnerability analysis on information, which may contain gaps that allow damage the integrity and reliability of web pages. With the results, it was possible to generate projections of problems, possible causes, and recommendations in order to alert and assist site administrators about potential problems, indicating a treatment near the ideal. To validate our proposal we performed experiments on real sites. The experimental results show the tool's effectiveness to identify different problems in all the analyzed sites, and indicate its possible solutions. Moreover, all the procedures were carried out correctly without damaging the structures and information from the pages of websites, making its use important to the reduction of existing attacks, helping developers and administrators of the sites that did not have knowledge of these faults or of the gravity that they can bring.
Keywords: web crawler, security, vulnerabilities, attacks, web pages