Segurança na Web: análise black-box de scanners de vulnerabilidades

Isadora Garcia Ferrão[1], **Diego Kreutz**[1]

[1]Universidade Federal do Pampa - Campus Alegrete (UNIPAMPA)
Av. Tiarajú, 810 - Ibirapuitã - 97.546-550 - Alegrete - RS - Brasil

isadora-gf@hotmail.com, Diego.Kreutz@unipampa.edu.br

Abstract. *Vulnerability scanners are essential tools for monitoring eminent security risks on Web systems. Unlike the existing literature, this paper presents a detailed black-box analysis of a significant set of free vulnerability scanners. The results allow to identify the main differences between existing scanners, contributing to a better selection of these important tools when performing vulnerability scans on Web systems.*

Resumo. *Os scanners de vulnerabilidades são ferramentas essenciais para monitorar riscos de segurança eminentes em sistemas Web. Diferentemente da literatura existente, este trabalho apresenta uma análise black-box detalhada de um conjunto significativo de scanners de vulnerabilidades gratuitos. Os resultados permitem identificar as principais diferenças entre scanners existentes, contribuindo para uma melhor seleção dessas importantes ferramentas na hora de realizar varreduras de vulnerabilidades em sistemas Web.*

1. Introdução

Sistemas *Web* fazem parte do cotidiano da maioria das pessoas. Segundo estatísticas, a cada ano milhares de novos sistemas são implementados e lançados na *Web* [Baldessar 2014]. Entretanto, apesar de a maioria desses sistemas possuírem um grande impacto social ou econômico, um conjunto significativo deles são lançados com vulnerabilidades críticas de segurança [Vieira et al. 2009]. A principal causa dessa alarmante constatação é o fato de os programadores se preocuparem essencialmente com as funcionalidades do produto final, pois possuem restrições de tempo para entrega do produto e não possuem os conhecimentos necessários sobre segurança de sistemas. Isto leva a um cenário propício à exploração maliciosa de dados e componentes de sistemas como comprovam relatórios de segurança atuais de grandes empresas, com equipes altamente especializadas em segurança de sistemas, como a Symantec e a Cisco.

Existem diferentes vulnerabilidades e ataques exploratórios que podem prejudicar os sistemas. Entre as mais comuns estão os ataques de injeção de código, *cross-site scripting* (XSS), quebra de autenticação e gerenciamento de sessão, referência insegura e direta a objetos, configuração incorreta de segurança, exposição de dados sensíveis, falta de função para controle do nível de acesso, *cross-site request forgery*, utilização de componentes vulneráveis conhecidos e redirecionamentos e encaminhamentos inválidos [OWASP 2017]. Na prática, os atacantes utilizam diferentes ferramentas para identificar e explorar estas vulnerabilidades dos sistemas, como *scanners* de vulnerabilidades disponíveis no mercado e *scripts* automatizados de teste e invasão.

Os *scanners* de vulnerabilidades são ferramentas especialmente desenvolvidas para diagnosticar defeitos de instalação, configuração e segurança de sistemas. Na engenharia de *software*, mais especificamente no ciclo de vida do *software*, este tipo de ferramenta é um instrumento que pode contribuir significativamente na automatização dos testes de robustez e segurança dos sistemas *Web*. Empresas e órgãos governamentais utilizam constantemente este tipo de ferramentas para detectar e corrigir vulnerabildades de sistemas. Como exemplo prático, a área de tecnologia da UNIPAMPA utiliza *scanners* de vulnerabilidades para testar periódicamente máquinas e sistemas. No momento em que vulnerabilidades são detectadas, usuários, desenvolvedores e especialistas de segurança são notificados para que as medidas apropriadas sejam tomadas.

A maioria dos *scanners* de vulnerabilidades de sistemas *Web* são usados como ferramentas para testes automatizados, de caixa preta, ou seja, sem acesso ao código-fonte, com o objetivo de identificar falhas e vulnerabilidades [Fong and Okun 2007]. As ferramentas de caixa preta simulam um ataque exploratório e fornecem relatórios contendo o local e as especificações das vulnerabilidades encontradas [Bau et al. 2010]. Essas ferramentas se tornaram tão importantes que *StartUps* estão sendo criadas com um único objetivo: prover varredura de vulnerabilidades para empresas e governos. Alguns exemplos são a On-security (`www.on-security.com`), Pentest-Tools (`pentest-tools.com`), Valency Networks (`www.valencynetworks.com`) e Veracode (`www.veracode.com`), especializadas em contratos de varredura de vulnerabilidades em sistemas *online*. Não apenas as empresas privadas, mas também instituições governamentais e forças armadas estão cada vez mais preocupados com a qualidade dos seus sistemas em termos de robustez e segurança, gerando contratos e oportunidades de negócio para empresas de segurança, em especial empresas especializadas em detecção de vulnerabilidades.

Apesar da existência de inúmeros *scanners*, que tem como objetivo detectar vulnerabilidades de sistemas *Web*, estudos demonstram que há disparidade entre as ferramentas existentes em termos de abrangência e níveis de exploração das vulnerabilidades [Rocha et al. 2012, Doupé et al. 2010, Vieira et al. 2009]. Os resultados apresentados na literatura indicam claramente a necessidade de não apenas uma única ferramenta, mas sim um conjunto de *scanners* para garantir uma boa cobertura na detecção de vulnerabilidades de sistemas *Web*. Outro aspecto importante de ser ressaltado é o fato de os estudos existentes serem relativamente antigos, isto é, não trazem o que há de mais recente em termos de ferramentas e técnicas de detecção de vulnerabilidades e, também, não explorarem a grande variedade e diversidade de ferramentas livres e abertas. Na verdade, alguns estudos exploram, em paticular, ferramentas comerciais.

Este trabalho tem como objetivo principal realizar uma análise *black-box* de *scanners* de vulnerabilidades livres e abertos. Para a análise foram selecionados dez dos principais *scanners* disponíveis na *Internet*, incluindo Uniscan, Paros Proxy, Zed Attack proxy, Nessus, Arachni, Grabber, Wapiti, Ratproxy, Skipfish e Vega [Rocha et al. 2012, Offensive Security 2017, INFOSEC 2017]. Estas ferramentas foram instaladas em uma máquina virtual Linux, ou sistema hospedeiro, e executadas sobre o mesmo alvo, uma máquina virtual do projeto BWA *(Broken Web Applications)* da OWASP [OWASP 2016]. No decorrer da análise comparativa das ferramentas, o trabalho procura responder três questões: *1- Quais vulnerabilidades são detectadas por cada scanner? 2- Qual a cober-*

tura de cada scanner? 3- Quais as vulnerabilidades mais frequentemente identificadas pelos scanners?

Os 10 *scanners* detectaram 28 vulnerabilidades nos sistemas da máquina virtual BWA. Entretanto, a média de vulnerabilidades detectadas pelos *scanners* foi baixa, isto é, aquém do esperado, com excessão do Zed Attack Proxy. Este *scanner* detectou 60.71% das vulnerabilidades, enquanto que o segundo melhor detectou apenas 35,71% e o pior deles ficou limitado a meros 3,57%. Estes resultados estão condizentes com outros trabalhos similares, ou seja, há discrepância significativa entre as ferramentas. Portanto, pode-se inferir que é imperativo avaliar os *scanners* de vulnerabilidades antes de decidir quais deles utilizar para manter os sistemas *Web* da empresa ou instituição livres de *bugs* passíveis de exploração por atacantes. No estudo realizado, a combinação dos *scanners* Zed Attack Proxy, Paros Proxy, Nessus, Skipfish e Vega oferece uma cobertura de 100% das vulnerabilidades presentes nos sistemas testados.

O restante deste texto está organizado como segue. A seção 2 apresenta a metodologia utilizada no processo de avaliação das ferramentas selecionadas. Na seção 3 são apresentados e discutidos os resultados empíricos dos testes realizados com *scanners* de vulnerabilidades. Por fim, são apresentados os trabalhos relacionados na seção 4 e a conclusão na seção 5.

2. Metodologia

Os *scanners* de vulnerabilidades tem evoluído significativamente com o passar do tempo. Da mesma forma, os testes realizados por estas ferramentas tem se tornado cada vez mais sofisticados. Como resultado, continuam sendo eficazes aliados na detecção de vulnerabilidades em sistemas.

O tipo mais comum de testes para *scanners*, também utilizado neste trabalho, é o caixa preta [Bau et al. 2010, Rocha et al. 2012, Doupé et al. 2010, Vieira et al. 2009, Perlman et al. 2016]. Os testes de caixa-preta são realizados de forma automatizada e sem acesso direto a detalhes de infraestrutura e codificação do sistema alvo.

Para realização dos testes, foram selecionadas as ferramentas Uniscan, Paros, Zed Attack Proxy, Nessus, Ratproxy, Grabber, Wapiti, Andiparos, Skipfish e Vega. Os principais motivos da escolha foram: serem gratuitas e de código aberto, não existirem estudos que comparem estas ferramentas, similaridade de objetivos entre elas e sites de classificação de ferramentas de segurança [Mundo Hacker 2017, Terminal Root 2017].

Os testes foram executados utilizando uma máquina Dell *Inspiron Special Edition*, Intel core i7, sexta geração, memória de 16 GB e 1 TB de HD + 8 GB de SSD. As ferramentas foram instaladas em uma máquina virtual (MV) com o sistema operacional Kali Linux versão 4.9.0. Como alvo para avaliar os *scanners*, foi utilizada uma máquina virtual OWASP BWA. Esta MV é constituída por um sistema operacional GNU/Linux, versão 1.2 e um conjunto de aplicativos *Web* (em diferentes linguagens de programação, como PHP, Perl e Python) vulneráveis especialmente preparados para testes de segurança.

3. Resultados

A tabela 1 sumariza os resultados obtidos com os testes de caixa-preta. As ferramentas são identificadas pelos seguintes índices: [1] Uniscan, [2] Paros, [3] Zed Attack Proxy,

[4] Nessus, [5] Ratproxy, [6] Grabber, [7] Wapiti, [8] Andiparos, [9] Skipfish e [10] Vega. Na tabela aparecem os nomes das vulnerabilidades, as ferramentas que a detectaram e o nível de gravidade da vulnerabilidade, respectivamente. O nível de gravidade foi definido conforme consta nos relatórios de saída dos *scanners*. Vale ressaltar que todos que detectaram uma determinada vulnerabilidade identificavam o mesmo nível de gravidade. Em outras palavras, não houve nenhuma discrepância entre os *scanners* em termos de criticidade das vulnerabilidades detectadas.

A maioria das vulnerabilidades são de criticidade alta ou média. As vulnerabilidades de criticidade alta representam 35,7% do total de vulnerabilidades detectadas. As vulnerabilidades com gravidade alta podem comprometer a confidencialidade, integridade e disponibilidade de um sistema. Da mesma forma, a mesma porcentagem foi alcançada pelas vulnerabilidades de criticidade média. Portanto, a soma das vulnerabilidades de gravidade média e alta perfaz 71,4% do total.

Para atacar uma vulnerabilidade do tipo mediana, o invasor precisa de acesso especializado, interação do usuário, ou outras circunstâncias que estão além do controle do invasor [Viegas 2016]. Uma vulnerabilidade de alta gravidade é mais fácil de ser explorada, pois não precisa de interação do usuário ou acesso especializado. De qualquer forma, é importante frizar que todas as vulnerabilidades de alta e média criticidade devem ser tratadas imediatamente para evitar incidentes, como vazamento de informações e indisponibilidade de sistemas.

As vulnerabilidades *external redirect, remote OS command injection, x-frame-options header not set, format string error, buffer overflow, x-content-type-options header missing, incorrecting caching directives, external content embedded on a page, unix operating system unsupported version detection, ssh weak algorithms supported, samba badlock vulnerability e shell injection* foram detectadas por apenas uma única ferramenta, como pode ser observado na tabela 1. Isto indica claramente a necessidade de múltiplos *scanners* de vulnerabilidades para garantir uma verredura mais completa e eficaz de sistemas *Web*.

É interessante observar que alguns *scanners*, de cobertura bastante baixa, como Nessus, Skipfish e Vega, detectaram vulnerabilidades específicas, que as demais ferramentas não detectaram. Já a ferramenta Zed Attack Proxy, além de detectar a maior porcentagem de vulnerabilidades, também detectou o maior número de vulnerabilidades de alta gravidade. Entretanto, sozinha ela não oferece uma cobertura completa (100% das vulnerabilidades) para o cenário da BWA. Mas, em conjunto com a Paros Proxy, Nessus, Skipfish e Vega, pode-se atingir uma cobertura de 100%.

A tabela 2 apresenta alguns resultados e métricas relativos aos processos de detecção de cada ferramenta, como a porcentagem em relação a todas as vulnerabilidades detectadas, o tempo de execução e a taxa de detecção das vulnerabilidades que possuem gravidade de nível alto. Como pode ser observado na tabela, o tempo de execução não tem a ver necessariamente com a cobertura. Os *scanners* Uniscan e Vega, apesar de possuirem um alto tempo de execução, não detectaram um número significativo de vulnerabilidades. Este elevado tempo de execução ocorre devido ao fato de estas ferramentas possuírem um número maior (e mais diversificado) de combinações de testes para determinadas vulnerabilidades, como *SQL injection*. Apenas as ferramentas Zed Attack e Skipfish detectaram

50% ou mais das vulnerabilidades de criticidade alta.

Tabela 1. Vulnerabilidades detectadas pelas ferramentas

Vulnerabilidade	Web Scanners	Gravidade
Sql Injection	[1,2,3,5,7,9,10]	Alto
Directory Browsing	[2,3,8,9,10]	Médio
Password Autocomplete in browser	[2,3,10]	Alto
Cross site scripting	[2,3,4,5,7,8,10]	Médio
Lotus Domino default files	[2,8]	Médio
Cross site scripting without brackets	[2,8]	Médio
Obsolete file	[2,8]	Baixo
Private ip disclosure	[2,3,8]	Baixo
Session ID in URL rewrite	[2,8]	Baixo
Obsolete file extended check	[2,8]	Baixo
Remote file inclusion	[3, 9]	Alto
Local file include	[1,3,7,9]	Alto
External Redirect	[3]	Alto
Remote OS Command Injection	[3]	Alto
Path Traversal	[3,4,9]	Alto
Application Error Disclosure	[3,4,10]	Médio
X-Frame-Options Header Not Set	[3]	Médio
Format string error	[3]	Médio
Buffer overflow	[3]	Médio
Web Browser XSS Protection Not Enabled	[3,9]	Baixo
Cookie No HttpOnly Flag	[3,4,6,9,10]	Baixo
X-Content-Type-Options Header Missing	[3]	Baixo
Incorrecting caching directives	[9]	Alto
External content embedded on a page	[9]	Baixo
Unix Operating System Unsupported Version Detection	[4]	Alto
SSH Weak Algorithms Supported	[4]	Médio
Samba Badlock Vulnerability	[4]	Médio
Shell injection	[10]	Alto

Vale também ressaltar que ferramentas como os *scanners* de vulnerabilidades também realizam ataques de DoS contra sistemas. A exemplo, uma dessas ferramentas foi utilizada recentemente para testar a robustez do Moodle da UNIPAMPA. O resultado do ataque foi uma indisponibilidade de aproximadamente 10 minutos do sistema. Somente após o endereço de origem do ataque ser bloqueado nos sistemas de segurança da instituição e o serviço ser re-estabelecido manualmente, o ataque deixou de ter efeito. Claro que, no caso, bastaria o atacante trocar de endereço IP para continuar o ataque.

No exemplo da UNIPAMPA, o problema está relacionado à implementação de um sistema moderno, amplamente utilizado e com vários ciclos evolutivos de melhoramento de *software*. Este tipo de ataque pode ser evitado através de técnicas e recursos de programação simples. Por exemplo, com a linguagem de programação PHP é possível verificar o tempo entre requisições subsequentes de forma simples. Em combinação com mapas do *ModRewrite* do servidor *Web* Apache, o sistema pode gerar, em tempo de execução, uma lista negra de IPs identificados como potenciais origens de ataques. O Apache irá utilizar os mapas para bloquear os ataques de DoS contra o sistema *Web*, o Moodle, no caso. Entretanto, é necessário que os engenheiros de *software* tenham formação apropriada em segurança de *software*, o que ainda não é algo comum, infelizmente.

Tabela 2. Principais resultados e métricas

Ferramenta	Porc. de deteção	Tempo de execução	Detecção gravidade alta
Uniscan	7,14 %	9 horas	20 %
Paros	35,71 %	1 hora e 03 minutos	20 %
Zed Attack	60,71 %	1 hora e 05 minutos	70 %
Nessus	21,43 %	15 minutos	20 %
Ratproxy	7,14 %	12 minutos	10 %
Grabber	3,57 %	5 minutos	0 %
Wapiti	10,71 %	2 horas	20 %
Andiparos	28,57 %	33 minutos	0 %
Skipfish	32,14 %	30 minutos	50 %
Vega	25,00 %	72 horas	30 %

Outra forma de conter ataques deste tipo é através de mecanismos avançados de tolerância a faltas e intrusões. Pesquisas e desenvolvimentos recentes tem mostrado que mecanismos como replicação de máquina de estados e componentes seguros podem ser utilizados para amenizar significativamente os efeitos de ataques de exaustão de recursos, ataques de DoS e intrusões que tenham como objetivo comprometer a confidencialidade e corretude do serviço [Kreutz et al. 2016].

4. Trabalhos Relacionados

Existem diferentes estudos que analisam e comparam *scanners* de vulnerabilidades [Bau et al. 2010, Vieira et al. 2009, Fong et al. 2008]. Estas pesquisas analisam ferramentas comerciais ou são consideravelmente desatualizadas. Em outras palavras, não há estudos recentes que analisam e comparam *scanners* de vulnerabilidades, em especial as opções livres e abertas.

Entre os *scanners* de vulnerabilidades avaliados na literatura, podem ser citados o *HP WebInspect* e o *IBM Rational AppScan*. Estes *scanners* se destacam pela boa cobertura de detecção de vulnerabilidades em cenários controlados.

Algumas das questões que os pesquisadores frequentemente procuram responder incluem: (1) Qual a cobertura dos *scanners* de vulnerabilidades quando usados em um ambiente de serviços *web*?, (2) Qual a taxa de falsos-positivos?, (3) Quais são os tipos mais comuns de vulnerabilidades em ambientes de serviços da *web*? e (4) Quais vulnerabilidades são as mais detectadas pelas ferramentas?. Neste trabalho foi respondido três das quatro questões. A única questão não respondida é a sobre falsos-positivos. A resposta a esta pergunta implicaria em analisar o código fonte de todas as dezenas de aplicações *Web* da máquina virtual BWA. Vale também ressaltar que falsos-positivos não ocorrem com tanta frequência em *scanners* de vulnerabilidades. Além disso, os falsos-positivos não representam um problema, uma vez que é melhor prevenir do que remediar. Em outras palavras, no pior caso falsos-positivos implicam em uma segunda verificação de propriedades, funcionalidades e códigos dos sistemas. Isto apenas ajuda a ratificar a corretude e segurança dos sistemas.

Os cenários mais comuns de avaliação de *scanners* de vulnerabilidades consistem em conjuntos limitados de serviços e sistemas controlados [Vieira et al. 2009]. Esse tipo de sistema é preparado especificamente para testes de segurança. Neste trabalho, foi utilizada a máquina virtual BWA, que possui uma boa coleção de aplicativos *Web* vulneráveis.

Esta máquina faz parte de um dos projetos da OWASP destinada a profissionais e pessoas interessadas em segurança e testes de *software*. Um dos pontos positivos da máquina virtual é que ela possui um conjunto significativo de vulnerabilidades.

Apesar de possuírem o mesmo intuito, os *scanners* detectam diferentes tipos de vulnerabilidades, com exceção da *SQL injection* e *cross site scripting* que, no geral, são detectadas por todas as ferramentas conhecidas. Os trabalhos relacionados detectam e analisam um conjunto limitado (em torno de 78% menos) de vulnerabilidades se comparados a este trabalho. Este cálculo foi baseado pela quantidade de vulnerabilidades detectadas. Enquanto que os trabalhos mencionados anteriormente informam a detecção de 6 vulnerabilidades, neste trabalho foram detectadas 28 vulnerabilidades distintas.

5. Conclusão

Neste trabalho foi analisada e discutida a eficácia de diferentes *scanners* de vulnerabilidades atuais, livres e abertos. Os resultados mostram a importância e a necessidade de se utilizar múltiplas ferramentas para encontrar e corrigir defeitos em sistemas *Web*. Utilizando um ambiente controlado, os *scanners* detectaram inúmeras vulnerabilidades de baixa, média e alta criticidade. É importante ressaltar que as vulnerabilidades de média e alta criticidade representam riscos de segurança críticos eminente para um sistema.

SQL injection e *cross site scripting* são as vulnerabilidades mais frequentemente testadas pelos *scanners*. Apesar disto, e do nível de criticidade, estas vulnerabilidades continuam na lista das mais recorrentes em sistemas *Web*. Este cenário é preocupante. As estatísticas permitem concluir que há uma falta de cuidados e utilização de ferramentas como *scanners* de vulnerabilidades no ciclo de vida de sistemas *online*.

Apesar de possuírem o mesmo intuito, as ferramentas detectam diferentes vulnerabilidades e diferentes níveis de gravidade, como é o caso do Nessus. Apesar do baixo índice de cobertura (21,43%), este *scanner* detectou vulnerabilidades que as outras ferramentas não reconheceram, como *samba badlock vulnerability, ssh weak algorithms supported e unix operating system unsupported version detection*. Outro exemplo é o Skipfish, que, apesar de apresentar uma cobertura de apenas 32,14%, detectou 50% das vulnerabilidades que representam risco de segurança alto.

A ferramenta mais eficaz foi a Zed Attack Proxy, com uma cobertura de 60,71% na detecção das vulnerabilidades da BWA. Entretanto, para atingir uma cobertura 100%, são necessários pelo menos mais quatro *scanners*, o Paros, Nessus, Skipfish e Vega. Estes resultados demonstram a necessidade de utilizar-se várias ferramentas de varredura de vulnerabilidades em sistemas *Web*. Um bom conjunto de *scanners* poderá oferecer uma boa cobertura na detecção de vulnerabilidades, diminuindo significativamente os riscos de comprometer a segurança dos sistemas.

Finalmente, é importante ressaltar que esta pesquisa terá continuidade através de diferentes ações e tarefas, como análise de testes específicos (exemplo: DoS) e utilização de diferentes ambientes controlados (para identificar taxas de falsos positivos e falsos negativos, por exemplo) e não controlados (sistemas disponíveis na *Web*). Outro objetivo futuro consiste em estudar *firewalls* de aplicação, ferramentas que podem ajudar a controlar ataques que visam explorar vulnerabilidades de sistemas *Web*.

Agradecimentos. Em especial ao Gustavo Alves Rodrigues pela contribuição técnica no que diz

respeito a mecanismos de defesa contra ataques de DoS em sistemas Web PHP/Apache.

Referências

Baldessar, M. J. (2014). Noticiário internacional: um mapa de contradições e influências ideológicas e econômicas1.

Bau, J., Bursztein, E., Gupta, D., and Mitchell, J. (2010). State of the art: Automated black-box web application vulnerability testing. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 332–345. IEEE.

Doupé, A., Cova, M., and Vigna, G. (2010). Why johnny can't pentest: An analysis of black-box web vulnerability scanners. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 111–131. Springer.

Fong, E., Gaucher, R., Okun, V., Black, P. E., and Dalci, E. (2008). Building a test suite for web application scanners. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 478–478. IEEE.

Fong, E. and Okun, V. (2007). Web application scanners: definitions and functions. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pages 280b–280b. IEEE.

INFOSEC (2017). 14 best open source web application vulnerability scanners. `https://goo.gl/DknrqB`.

Kreutz, D., Malichevskyy, O., Feitosa, E., Cunha, H., da Rosa Righi, R., and de Macedo, D. D. (2016). A cyber-resilient architecture for critical security services. *Journal of Network and Computer Applications*, 63(Supplement C):173 – 189.

Mundo Hacker (2017). Ferramentas para scan de vulnerabilidades web. `https://goo.gl/VNVcnP`.

Offensive Security (2017). Kali tools - web applications. `https://goo.gl/QfvN4u`.

OWASP (2016). Broken web applications project. `https://goo.gl/NScjJ`.

OWASP (2017). Top ten 2017 project. `https://goo.gl/snkFmd`.

Perlman, R., Kaufman, C., and Speciner, M. (2016). *Network security: private communication in a public world*. Pearson Education India.

Rocha, D., Kreutz, D., and Turchetti, R. (2012). A free and extensible tool to detect vulnerabilities in web systems. In *Information Systems and Technologies (CISTI), 2012 7th Iberian Conference on*, pages 1–6. IEEE.

Terminal Root (2017). 100 melhores ferramentas open source de segurança. `https://goo.gl/9ksoXR`.

Viegas, J. (2016). Por que o número de ataques virtuais aumentou e como evitá-los. `https://goo.gl/zkNahP`.

Vieira, M., Antunes, N., and Madeira, H. (2009). Using web security scanners to detect vulnerabilities in web services. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*, pages 566–571. IEEE.