View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Archives of the Faculty of Veterinary Medicine UFRGS

INVESTIGAÇÃO DE SEGURANÇA NO MOODLE

Rodrigo Ronner Tertulino da Silva, Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte - IFRN, Campus Apodi, Rio Grande do Norte.
rodrigo.tertulino@ifrn.edu.br
Rommel Wladimir de Lima, Universidade do Estado do Rio Grande do Norte - UERN, Mossoró, Rio Grande do Norte. rommel.lima@gmail.com
Cicilia Raquel Maia Leite, Universidade do Estado do Rio Grande do Norte - UERN, Mossoró, Rio Grande do Norte.
cicilia.maia@gmail.com
Romero Rossano Tertulino da Silva, Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte - IFRN, Campus Parnamirim, Rio Grande do Norte.
romero.tertulino@ifrn.edu.br

RESUMO

O Moodle é uma aplicação Web definida como uma plataforma de educação a distância usada em muitas universidades como forma de permitir a interação do aluno *on-line*. Instituições fazem uso do Moodle por causa da sua flexibilidade, adaptabilidade e facilidade de uso. De modo que, possui uma substancial base de usuários com 56.185 sites ativos e 46.343.749 usuários finais. Este artigo tem como objetivo descrever as falhas de segurança encontradas através da aplicação de técnicas de exploração de vulnerabilidades. O Moodle foi escolhido para validação por ser a principal aplicação de código aberto utilizada nos dias de hoje para prover educação a distância.

Palavras-chaves: Hacker. Moodle. Penetração. Segurança. Vulnerabilidades.

SAFETY INVESTIGATION ON MOODLE

ABSTRACT

Moodle is a web application set as a platform for distance education used in many universities as a way to allow interaction of the online learner. Moodle institutions make use because of its flexibility, adaptability and ease of use. So, have a substantial user base with 56.185 active sites and 46,343,749 end users. This article aims to describe the security flaws found by applying techniques of exploitation of vulnerabilities. Moodle was chosen for validation due to be a major open source application used today to provide distance education.

Keywords: Moodle. Security. Hacking. Penetration. Vulnerabilities.

1. Introdução

Com a crescente globalização e com a utilização em larga escala da internet, cada vez mais interativa, os usuários se veem cercados de facilidades no mundo digital. Hoje os sites estão cada vez mais dinâmicos e interativos, gerando, assim, uma troca de informações entre servidores e usuários. É nessa troca de informações que *hackers*

podem se aproveitar e acessar informações em bancos de dados de servidores de aplicações web. Por isso, uma aplicação vulnerável se torna alvo fácil para os *hackers* profissionais (JACOBS, 2011).

Os ataques estão cada vez mais utilizando métodos automatizados de exploração de vulnerabilidades. Segundo dados do Cert.br,[1] de janeiro a dezembro de 2013, 24% dos incidentes reportados no Brasil tinham como foco tentativas de fraudes, 5% estavam direcionadas a aplicações Web (CERT.BR, 2014).

O Moodle provou a sua importância devido a uma maior aceitação na comunidade e ao número de instituições que fazem uso dele. Bem como, fornece suporte para um grande número de cursos em diferentes idiomas (COLE *et al*. 2007). Ele permite aos usuários a possibilidade de postar notícias, atribuições, revistas eletrônicas e adicionar recursos.

Com crescimento da comunidade em torno do projeto; desenvolvedores e usuários participam dos fóruns, discutindo e compartilhando dicas, postando trechos de código, ajudando aos novos usuários, compartilhando recursos e debatendo novas ideias que enriquecem o Moodle (BARRINGTON, 2014).

Para a realização dos testes de penetração, as vulnerabilidades que foram consideradas nos testes são elencadas pelo OWASP, este detalha as dez vulnerabilidades mais comuns em aplicações Web (OWASP, 2013).

No sentido de melhorar a segurança do Moodle, o presente trabalho sugere ações que podem ser colocadas em prática pela equipe de desenvolvimento como forma de mitigar os problemas que foram descobertos, contribuindo para a seguridade da aplicação.

Diante do exposto, o objetivo deste trabalho é realizar testes de penetração direcionados ao Moodle, alertando aos usuários e profissionais que fazem uso desse sistema a fim de verificar a segurança e, ao mesmo tempo, propor soluções.

2. A Necessidade da Educação a Distância

A Educação a Distância (EAD) é a modalidade educacional na qual alunos e professores estão separados, física e/ou temporalmente e, nessa interação, eles fazem uso de Tecnologias de Informação e Comunicação (TIC). Essa modalidade é regulada por uma legislação específica e pode ser implantada na educação básica (educação de jovens e adultos, educação profissional técnica de nível médio) e na educação superior (MEC, 2014).

A EAD vem crescendo intensamente no Brasil e no mundo. Em 2012, de acordo com dados do Censo de Educação Superior, dos 6,7 milhões de universitários brasileiros, 14,7% estavam matriculados em cursos a distância. Já no Canadá, país pioneiro na massificação da EAD, seus 32 milhões de cidadãos têm à disposição 56 universidades, das quais 53 oferecem cursos a distância (EAD, 2014).

Para atender a esta demanda educacional, os Ambientes Virtuais de Aprendizagem (AVAs) estão sendo cada vez mais utilizados na EAD como uma opção

[1] Centro de estudos, resposta e tratamento de incidentes de segurança no Brasil.

tecnológica. Em termos conceituais, os AVAs consistem em um agrupamento de recursos tecnológicos resultantes da evolução das TIC que permitem a emissão e recepção de mensagens que utilizam o ciberespaço para vincular conteúdo e permitir interação entre atores do processo educativo. Porém, a qualidade do processo educativo depende do envolvimento do aprendiz, da proposta pedagógica, dos materiais veiculados, da estrutura e qualidade de professores, tutores, monitores e equipe técnica e também das ferramentas e recursos tecnológicos utilizados no ambiente (PEREIRA, 2007).

O Moodle foi lançado em 2001, como uma nova alternativa, ele é definido através da sigla, em inglês, *Modular Obejct-Oriente Dynamic Learning Environment* que significa Ambiente Modular de Aprendizagem Dinâmica Orientada a Objetos. Ressalta-se que esta plataforma encontra-se em constante desenvolvimento por milhares de programadores espalhados por todos os lugares do mundo, oferecendo também suporte aos usuários, novas funcionalidades e tradução para vários idiomas (AL-AJLAN, 2012).

3. Vulnerabilidades em Aplicações Web

O *Open Web Application Security Project* (OWASP) é uma entidade sem fins lucrativos e de reconhecimento internacional e contribui para a melhoria da segurança de softwares aplicativos reunindo informações importantes que permitem avaliar riscos de segurança e combater formas de ataques através da internet (OWASP, 2013).

De acordo com estudos recentes do OWASP, é possível destacar o conjunto das dez vulnerabilidades mais exploradas em aplicações Web (HAROLD, 2010).

Tabela 1 - Vulnerabilidades Conhecidas em Aplicações Web Fonte: Adaptado de HAROLD (2010)

Casos de Abuso	Risco
Injeção de Código	Ocorrem quando dados não confiáveis são enviados para um interpretador como parte de um comando ou consulta. Os dados manipulados pelo atacante podem iludir o interpretador para que este execute comandos indesejados ou permita o acesso a dados não autorizados.
Quebra de Autenticação e Gerenciamento de Sessão	Ocorre quando as funções da aplicação relacionadas à autenticação e ao gerenciamento de sessão são implementadas de forma incorreta, permitindo que os atacantes comprometam senhas, chaves e *tokens* de sessão ou explorem outra falha da implementação para assumir a identidade de outros usuários.
Cross-Site Scripting (XSS)	Falhas XSS permitem aos atacantes executarem *scripts* no navegador da vítima.
Referência Insegura e Direta a Objetos	Ocorre quando um programador expõe uma referência à implementação interna de um objeto, como um arquivo, diretório, ou registro da base de dados.
Configuração Incorreta de Segurança	Uma boa segurança exige a definição de uma configuração segura em todos os níveis.
Exposição de Dados Sensíveis	Ocorre quando as aplicações não protegem devidamente os dados sensíveis, tais como cartões de crédito, IDs fiscais e credenciais de autenticação.
Falta de Função para Controle do Nível de Acesso	Ocorre quando as aplicações não verificam os direitos de acesso, em nível de função, antes de tornar essa funcionalidade

	visível na interface do usuário.
Cross-Site Request Forgery (CSRF)	Ocorre quando o navegador da vítima é forçado a executar uma ação maliciosa em favor do atacante.
Utilização de Componentes Vulneráveis Conhecidos	Componentes, tais como bibliotecas, *frameworks*, e outros módulos de *software* quase sempre são executados com privilégios elevados.
Redirecionamentos e Encaminhamentos Inválidos	Aplicações frequentemente redirecionam e encaminham usuários para outras páginas ou sites e usam dados não confiáveis para determinar as páginas de destino.

4. Metodologia para Testes de Penetração

Os Testes de Penetração correspondem a uma técnica que procura realizar uma tentativa de invasão legal. Com o intuito de explorar uma vulnerabilidade existente em uma aplicação e evidenciando os problemas aos quais uma determinada aplicação está suscetível, possibilitando, assim, que as falhas descobertas através da realização dos testes, possam ser resolvidas antes que uma pessoa maliciosa consiga explorá-las. Desse modo, essa metodologia consiste em fazer uso de ferramentas e artifícios utilizados por um invasor (ENGEBRETSON, 2013).

Segundo Engebretson (2013), a metodologia para técnicas de penetração é dividida em quatro etapas: Reconhecimento, *Scanning*, Exploração de falhas (*exploitation*) e Pós-Exploração (ou Preservação do Acesso). Entender os passos necessários para realização de um teste de penetração é imprescindível para que se possa realizar de um teste de penetração amplo e efetivo. A figura 1 apresenta essas atividades.

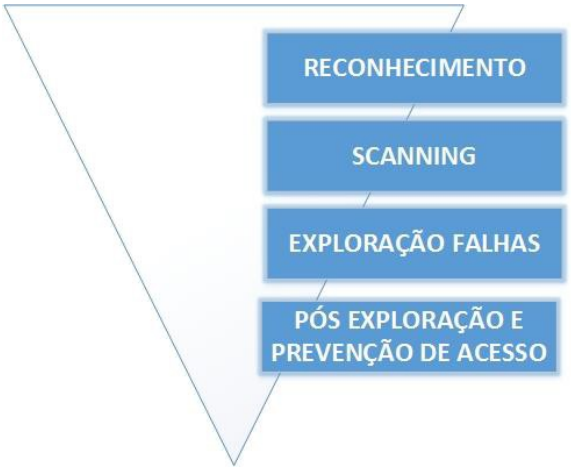

Figura 1 - A metodologia Hacking para testes de penetração
Fonte: (Engebretson, 2013)

O primeiro passo na metodologia é denominado reconhecimento, tem o propósito de colher o máximo de informações sobre o alvo (ENGEBRETSON, 2013). De modo que, quanto maior a quantidade de informações coletadas sobre o alvo que se pretende explorar, maior a probabilidade desse reconhecimento ser útil nas próximas etapas.

A segunda etapa nessa metodologia pode ser particionada em duas atividades diferentes. A primeira atividade a ser realizada é denominada *scanning* de portas. Trata-se de realizar uma varredura com o objetivo de descobrir os serviços e portas abertas no alvo pretendido, logo em seguida após concluir o *scanning* de portas, será criada uma lista de portas abertas e de serviços em potencial sendo executados no alvo. A segunda

atividade da fase de *scanning* é o *scanning* de vulnerabilidades. Esse *scanning* de vulnerabilidades tem como objetivo de localizar e identificar pontos de vulnerabilidades (falhas) existentes em aplicações que estão sendo executadas no alvo (ENGEBRETSON, 2013).

Com os resultados obtidos na segunda etapa, o próximo passo será a fase de exploração de falhas (*exploitation*). Após a descoberta de quais portas e de quais serviços estão sendo executados, poderá ser iniciado um ataque direcionado ao alvo que se pretende explorar (ENGEBRETSON, 2013).

A etapa final a ser realizada é a de pós-exploração e preservação do acesso. Uma aplicação web faz uso do protocolo HTTP (*Hypertext Transfer Protocol*), o protocolo tem como característica ser sem estado (*stateless*), ou seja, a cada requisição e resposta é criada uma nova sessão. Neste sentido, se faz necessário prosseguir para a fase de pós-exploração para criar um *backdoor*[2], possibilitando que o acesso esteja disponível para futuros acessos (ENGEBRETSON, 2013).

Primeiro foi realizado reconhecimento, coletando as informações sobre o Moodle, logo em seguida foi realizado s*canning* com as ferramentas especificas em face às vulnerabilidades elencadas no OWASP *TOP* 10, onde foram feitas tentativas de acesso, a fim de se descobrir as falhas existentes no Moodle, E por fim, foi feita a exploração dessas falhas que foram descobertas na etapa de s*canning*.

A atividade de preservação do acesso incluída na última etapa não foi necessário, devido não haver necessidade de manter acesso para futuras invasões, como sugere a própria metodologia de penetração, o propósito era apenas de descobrir a quais falhas o Moodle está suscetível.

As ferramentas para realização dos testes de penetração são todas de código fonte aberto, tais ferramentas listadas na tabela 2 fazem parte do *Kali Linux* (KALI, 2014). A primeira coluna apresenta o nome das ferramentas que foram utilizadas. A segunda coluna apresenta a sua respectiva descrição e o propósito de cada uma na realização de um teste de penetração.

Tabela 2 – Apresenta as Ferramentas Utilizadas para Realização de Testes de Penetração.

Ferramenta	Descrição
Metasploit Framework (MSF)	Tem como objetivo realizar análise de vulnerabilidades de segurança e facilitar testes de penetração. Utilizada nos testes de Quebra de autenticação.
Nikto	Trata-se de um scanner que realiza testes abrangentes contra servidores Web. Também verifica quais são os itens de configuração do servidor, tais como a presença de arquivos de índice, opções do servidor HTTP, tentando identificar os servidores web instalados e quais *softwares* fazem parte. Utilizada nos testes de Referência insegura e direta a objetos.
Web Application Attack and Audit Framework (W3af)	É um *scanner* de segurança de aplicações web de código aberto. Fornece informações sobre vulnerabilidades de segurança e ajuda no esforço de teste de penetração. Utilizada nos testes de Injeção de código *Sql*.

[2] *Backdoor* é um recurso utilizado por diversos *malwares* para garantir acesso remoto ao sistema ou à rede infectada, explorando falhas críticas não documentadas existentes em programas instalados, *softwares* desatualizados e do *firewall* para abrir portas do roteador.

Spidering	São principalmente utilizados para criar uma cópia de todas as páginas visitadas para um pós-processamento por um motor de busca que irá indexar as páginas baixadas para prover buscas mais rápidas. *Crawlers* também podem ser usados para tarefas de manutenção automatizadas em um *website*, como checar os links ou validar o código HTML. Utilizada nos testes de Utilização de Componentes Vulneráveis Conhecidos.
WebScarab	Ferramenta de teste de aplicações de segurança Web. Ele serve como um *proxy* que intercepta e permite que as pessoas alterem pedidos feitos a navegadores web (HTTP e HTTPS) e respostas do servidores Web. Utilizada nos testes de *Cross-Site Request forgery (CSRF).*
OWASP-ZAP	É uma ferramenta utilizada em testes de penetração desenvolvida para encontrar vulnerabilidades em aplicações Web. Fornece *scanners* automatizados, bem como um conjunto de ferramentas que permitem encontrar vulnerabilidades de segurança manualmente. Utilizada nos testes de Redirecionamentos e Encaminhamentos Inválidos, Configurações incorretas de segurança, Exposição de dados sensíveis, Falta de função para controle do nível de acesso e *Cross-Site Scripting* (XSS).

5. Configuração do Ambiente e Resultados dos Testes de Penetração

Foram feitos testes de penetração, utilizando ferramentas apropriadas. A tabela 3 apresenta todos os aplicativos utilizados nos testes e os equipamentos que foram necessários. A primeira coluna diz respeito aos equipamentos, a segunda coluna aos *softwares* que foram utilizados e a terceira e última à finalidade.

Tabela 3 – Cenários dos Testes Realizados

Máquina	Configuração	Finalidade
Servidor Físico DELL T410	•*Linux Centos* 7.0 •Moodle 2.7.1 •Apache/2.4.6 •PHP/5.4.16 •Maria DB 5.5.37 •HTTP/1.1	Esta máquina atua como Servidor do Moodle em ambiente *Linux*.
Máquina Virtual 01	•*Kali Linux*	*Kali Linux* é uma distribuição Linux baseada em Debian para realização de testes de penetração.
Máquina Virtual 02	•*Ubuntu* 14.04.1 LTS	Utilizado como máquina cliente para acesso ao Moodle através da rede.
Notebook Dell Vostro	•*Windows* 8.1 64 bits •*VirtualBox* 4.3.14 for	Utilizado como máquina onde será instalado Virtualbox.

Uma série de testes foi realizada utilizando a configuração, conforme ilustrada na tabela 4, a seguir. A primeira coluna refere-se ao nome das vulnerabilidades que foram analisadas, na segunda coluna são informados os *status* quanto à descoberta, ou não, de alguma vulnerabilidade. Na terceira coluna é possível visualizar de forma quantitativa as falhas que foram descobertas em face da aplicação das ferramentas elencadas na quarta coluna.

Não foram encontradas vulnerabilidades quanto às falhas de segurança que incluem Injeção de Código Sql, Referência Insegura e Direta a Objetos, *Cross-Site Request forgery* (CSRF), Utilização de Componentes Vulneráveis Conhecidos, Redirecionamentos e Encaminhamentos Inválidos.

Tabela 4 – Resumo dos Resultados

Vulnerabilidade	Status	Qtd	Ferramenta
Injeção de código *Sql*	Não Encontrada	0	W3af
Referência insegura e direta a objetos	Não Encontrada	0	Nikto
Cross-Site Request forgery (CSRF)	Não Encontrada	0	WebScarab
Utilização de Componentes Vulneráveis Conhecidos	Não Encontrada	0	Spidering
Redirecionamentos e Encaminhamentos Inválidos	Não Encontrada	0	OWASP Zap
Quebra de autenticação	Encontrada	272	Metasploit
Configurações incorretas de segurança	Encontrada	242	OWASP Zap
Exposição de dados sensíveis	Encontrada	4	OWASP Zap
Falta de função para controle do nível de acesso	Encontrada	11	OWASP Zap
Cross-Site Scripting (XSS)	Encontrada	2	OWASP Zap

No que diz respeito à quebra de autenticação foram encontradas um total de 272 (duzentos e setenta e duas) no Moodle. Essas quebras foram geradas devido a não utilização da *flag HttpOnly*. A *flag* mitiga as ações de atacantes quando estes tentam realizar sequestro de sessão através de *CSRF*. A *flag HttpOnly* ajuda a reduzir o problema quanto à quebra de autenticação, mas para que isso ocorra é necessário que o navegador do usuário suporte a *flag*. Atualmente a maioria dos navegadores possui suporte a *Httponly* (PAULI, 2014). O seu uso é indicado no corpo das respostas das requisições HTTP. Conforme exemplo abaixo.

Reposta HTTP 1.1:

HTTP/1.1 200 OK
Date: Sat, 02 Aug 2014 22:13:16 GMT
Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod_auth_kerb/5.4 mod_fcgid/2.3.9 mod_nss/2.4.6 NSS/3.15.4 Basic ECC PHP/5.4.16 mod_wsgi/3.4 Python/2.7.5
X-Powered-By: PHP/5.4.16
Set-Cookie: user=t=bfabf0b1c1133a822; path=/; **HttpOnly**

Foram encontradas 242 (duzentas e quarenta e duas) configurações incorretas de segurança. Isso é ocasionado pela falta de configuração no cabeçalho *X-Content-Type-Options*, onde a opção *Type-Options* não foi definida como *nosniff*. Esse resultado permite que versões mais antigas do Internet Explorer e Chrome possam executar *MIME-Sniffing* no corpo da resposta, podendo causar respostas no corpo a ser interpretadas e exibidas como um tipo de conteúdo que não seja o tipo declarado. Portanto, se faz necessário setar a opção *Type-Options* como *nosniff* para mitigar problemas relacionados às configurações incorretas de segurança.

Quanto aos problemas com exposição de dados sensíveis foram encontrados um total de 4 (quatro) formulários de preenchimento dentro do Moodle. A função AUTOCOMPLETE não está desabilitada, sendo assim, senhas podem ser armazenadas e depois recuperadas. Por isso é altamente aconselhável deixar a função AUTOCOMPLETE nos formulário contendo senhas AUTOCOMPLETE=*'OFF'*.

No que diz respeito à falta de função para controle do nível de acesso foram encontradas um total de 11 (onze) vulnerabilidades, isso se deve ao fato de *X-Frame-*

Options header não está incluído na resposta HTTP para proteger contra ataques *clickjacking*. Ataques do tipo *clickjacking* estão relacionados às formas de enganar o usuário. E desse modo, o usuário pensa que está realizando uma operação em um determinado site, quando na verdade suas ações estão sendo executadas em outra página.

Foram detectadas 02 (duas) vulnerabilidades no que diz respeito à XSS. A XSS é a vulnerabilidade mais disseminada em aplicações web atualmente (RANSOME, 2013), porém, com frequência, é menosprezada. Ao acessar um site, o navegador estabelece uma relação de confiança com o site (PAULI, 2014). Após ser estabelecida uma relação de confiança entre o site e o usuário, a conexão passa a ser vista para aplicação como sendo confiável, visto que uma relação de confiança foi estabelecida anteriormente.

Na figura 2 é apresentada a execução de um código escrito em *javascript* evidenciando a vulnerabilidade.

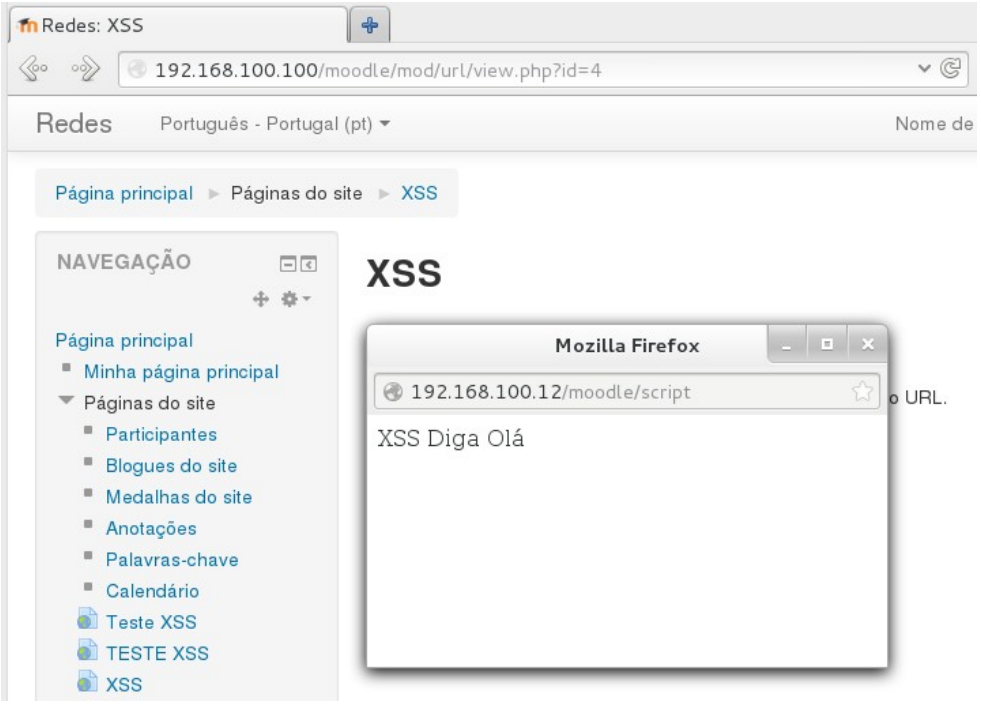

Figura 2 - Execução código em *javascript* por meio de uma vulnerabilidade.

Como forma de contornar esse problema, a solução mais prática seria incluir no cabeçalho HTTP a opção *X-Frame-Options*, marcado com *"SAMEORIGIN."*. Dessa forma, permitiria apenas que a página pudesse ser mostrada em um quadro que venha da mesma origem que a própria página.

Umas das principais consequências da vulnerabilidade de XSS é a exposição negativa do sistema e a possibilidade de utilização da falha para a distribuição de *phishing*[3] e facilitação de fraudes.

[3] É uma forma de fraude eletrônica, utilizada com o intuito de obter dados pessoais de diversos tipos, dentre estes: senhas, informações financeiras como número de cartões de crédito e outros dados pessoais.

6. Considerações Finais e Trabalhos Futuros

O desenvolvimento do Moodle ao longo do tempo foi feito em partes e à medida que o software se tornava cada vez mais utilizado, ocasionava uma necessidade por novas funcionalidades.

Entre as inúmeras vulnerabilidades, estudos recentes realizados pela OWASP (2013) indicaram as dez mais recorrentes e mais exploradas pelos invasores. Muitas destas vulnerabilidades foram identificadas pela simples realização de atividades de testes de penetração

Neste sentido, é importante ressaltar ainda que a segurança do Moodle deve ser estudada e levada a sério, pois problemas como os reportados podem não só comprometer as instituições que implementaram o Moodle, mas também o público que utiliza esta plataforma de educação a distância.

Nos testes aplicados é perceptível que a maioria das vulnerabilidades está relacionada às tentativas e técnicas que procuram enganar os usuários, como é o caso da vulnerabilidade por XSS, mas que devem ser mitigadas pela equipe de desenvolvimento, visto que o usuário é a parte mais fraca do sistema.

O presente trabalho teve como intuito maior evidenciar falhas existentes na versão mais recente do Moodle 2.7, alertando a comunidade sobre os problemas de segurança aos quais essa aplicação está suscetível. Desse modo, foram demonstradas técnicas de exploração de falhas para que outras pessoas possam utilizá-la e realizar seus próprios testes, destacamos também quais ferramentas são úteis na realização dos testes de penetração.

Como trabalhos futuros, sugerem-se o desenvolvimento de um framework especializado que realize os testes de vulnerabilidade de forma dinâmica em uma única aplicação e também o desenvolvimento de uma metodologia que contemple a inclusão de etapas de segurança no ciclo de desenvolvimento de um software, mitigando problemas de vulnerabilidade no desenvolvimento de um software, tornando-o mais seguro.

Referências

Al-Ajlan, A. S. A. comparative study between e-learning features, methodologies, tools and new developments for e-learning. dr. elvis pontes (ed.). In: **Methodologies, Tools and New Developments for E-Learning. INTECH**, 2012. p. 25. Qassim University Kingdom of Saudi Arabia. Disponível em: < http://www.intechopen.com/books/methodologies-tools-and-new-developments-for-e-learning/a-comparative-study-between-e-learning-features>.

Barrington, Rebecca. **Moodle Gradebook - Set up and customize the gradebook to track student progress through Moodle**. Packt Publishing, BIRMINGHAM – MUMBAI, 2012.

CERT.BR – **Centro de Estudos, Respostas e Tratamentos de Incidentes de Segurança no Brasil**, 2014. Disponível em: <http://www.cert.br/stats/incidentes/2013-jan-dec/tipos-ataque.html>. Acesso em: 15/09/2014.

Cole, J. And Foster, H., **Using Moodle: Teaching with the Popular Open Source Course Management System**, 2nd ed. O'Reilly, 2007.

EAD – **Seu Portal de Ensino a distância**" 2014. Disponível em: <http://www.ead.com.br/expansao-ead-brasil/>. Acesso em: 01/10/2014.

Engebretson, Patrick. **Basics of hacking and penetration testing, the ethical hacking and penetration testing made easy**., SYNGRESS (ELSEVIER), 2013.

Harold F. Tipton. 2010. ***Official (Isc)2 Guide to the SSCP Cbk*, Second Edition (2nd ed.)**. Auerbach Publications, Boston, MA, USA.

Jacobs, Stuart. **Engineering information security: The application of systems engineering concepts to achieve information assurance**. p. cm. ISBN 978-0-470-56512-4 (hardback). July 2011, Wiley-IEEE Press.

KALI LINUX – **Free all-in-one solution for professional security auditing**, 2014. Disponível em: < http://www.kali.org/>. Acesso em: 15/09/2014.

Kumar, S.; Gankotiya, Ak.; Dutta, K., A comparative study of moodle with other e-learning systems. **Electronics Computer Technology (ICECT)**, 2011 3rd International Conference on , vol.5, no., pp.414,418, 8-10 April 2011 doi: 10.1109/ICECTECH.2011.5942032 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5942032&isnumber=5941942

Lakhan, Shaheen E., AND KAVITA JHUNJHUNWALA. **Open Source Software in Education**. EDUCAUSE QUARTERLY 2 Nov. 2008: 33-40. Web.

MEC – **Ministério da Educação**, 2014. Disponível em http://www.mec.gov.br/. Acesso em: 12/07/2014.

OWASP, OWASP Top Ten – 2013. ***The Ten Most Critical Web Application Security Risks***. Disponível em https://www.owasp.org/images/9/9c/OWASP_Top_10_2013_PT-BR.pdf. Acesso em: 05/07/2014.

Pauli, Josh. **Introdução ao Web Hacking: Ferramentas e técnicas para invasão de aplicações web**. São Paulo, SP: Editora Novatec,. Janeiro de 2014.

Pereira, Alice T. Cybis. **Ambientes Virtuais de Aprendizagem – Em diferentes Contextos**. Rio de janeiro: Editora Ciência Moderna., 2007.

Pontes, Elvis., Silva, Anderson., Guelfi, Adilson., Kofugi, Sérgio. Methodologies, Tools and New Developments for E-Learning. ISBN 978-953-51-0029-4, 332 pages, Publisher: **InTech, Chapters published February** 03, 2012 under CC BY 3.0 license DOI: 10.5772/1115.

Ransome, James And Misra, Anmol. 2013. **Core Software Security: Security at the Source**. Auerbach Publications, Boston, MA, USA.