Auditoria em tecnologia de informação: análise das ameaças à segurança de ambientes web na Unoesc.

Alan Zarpelon Bernardi *
Fabiano de Oliveira Wonzoski **

Resumo

A Tecnologia da Informação(TI) é uma área fundamental para que uma organização seja competitiva no mercado. A Unoesc como instituição de ensino procurou nos últimos anos automatizar a maior quantidade de rotinas possíveis, evoluindo consideravelmente e atendendo níveis mais altos dentro da sua hierarquia, passando a oferecer recursos para tomada de decisão pelos gestores. Tal evolução possibilitou que os dados totalmente em formato digital se tornassem ferramenta para o alcance de metas e objetivos planejados pela instituição. A dependência e importância dos dados computacionais são itens fundamentais, quando relacionados a necessidade de uma auditoria em ambiente de TI. Dessa forma, a auditoria em meio a esse ambiente altamente mutável, representa uma maneira de garantir a integridade e segurança dos dados. Aplicada mediante uma metodologia e seguindo normas e políticas internas, visa medir os riscos, buscar falhas, atestando a segurança do ambiente objeto de estudo. Se tratando de ambientes web, as aplicações estão expostas a inúmeros riscos que podem comprometer sua estrutura. Uma boa técnica para se validar a segurança em aplicações é por meio de testes de invasão onde, utilizando ferramentas de análise, é validada a solidez do ambiente. O estudo o qual se trata este artigo, contemplou testes de invasão e análise de resultados, utilizando ferramentas que fazem parte da distribuição Kali Linux.

Palavras-chave: Kali. Invasão. Segurança da informação. Web.

Abstract

Audit in information technology: analysis of threats to the security of web environments at Unoesc.

Information Technology(IT) is a fundamental area for an organization to be competitive in the marketplace. Unoesc as an educational institution has sought in recent years to automate as many routines as possible in order to evolve considerably and attend higher levels within the company hierarchy, offering resources for decision making by managers. This evolution allowed the data in digital format to become a fundamental tool to the institution objective and goal achievement. The dependence and significance of the computational data are fundamental when related to the need of an audit in the IT environment. Therewith, the audit emerges in this highly changeable environment as a way to ensure data integrity and security. It aims to measure the risks, look for flaws and, attest the safety of the environment under study by applying methodology and following internal norms and policies. When it comes to web environment, risks can expose applications and compromise its structure. A good technique to validate an application security is through invasion tests that uses analysis tools to solidify the integrity of the environment. This study presents invasion tests and results analysis using tools that are part of the Kali Linux distribution.

Keywords: Kali. Invasion. Informationsecurity. Web.

1 INTRODUÇÃO

O avanço da Tecnologia da Informação(TI) dentro do ambiente organizacional tem colocado em evidência o valor que o bem informação tem em relação aos objetivos da empresa. Em contraponto, busca-se proteger e garantir a segurança para sua efetiva utilização com a finalidade de atender as necessidades de gestores.

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

Segundo Fontes (2006), define-se como segurança da informação o conjunto de orientações, normas, procedimentos, políticas e demais ações para que o recurso informação esteja protegido.A segurança da informação existe com a finalidade de diminuir o risco que um determinado negócio apresenta em relação à dependência do seu uso para o funcionamento da organização.

De acordo com Rorrato e Dias (2014), cuidados devem ser tomados de tal forma que se possa verificar a integridade e garantir que os dados estejam protegidos. A auditoria constitui-se como uma forma eficiente de manter um ambiente seguro.

Segundo Lento e Guimarães (2012), em termos gerais a auditoria pode ser compreendida como a atribuição responsável pela fiscalização dos processos, tendo como função verificar se eles estão sendo executados de forma constante e correta e se os mesmos são independentes. Aplicada seguindo métodos preestabelecidos, a auditoria verifica e garante que o objeto auditado está de acordo com o estipulado.

Na área de TI, todos os processos podem e devem ser auditados, desde a decisão sobre tecnologias a serem adotadas, desenvolvimento de sistemas, integração entre sistemas, comunicação entre máquinas, mudanças de sistemas e tecnologias, equipe de desenvolvimento, prioridades, controles organizacionais, legalidade jurídica, bem como os resultados. A auditoria no setor TI é imprescindível, haja vista que, hoje, muitas organizações param de operar pelo fato de a tecnologia de TI adotada deixar de funcionar.

Com base no conceito de auditoria em Tecnologia da Informação e tendo em vista a sua importância, este artigo constitui-se de uma revisão bibliográfica voltada aos principais aspectos de segurança em ambientes web, através da análise de vulnerabilidades encontradas e impactos gerados pela inoperabilidade de seus serviços. A partir da revisão, foi elaborada uma pesquisa experimental, através da qual se propôs as adequações necessárias para a correção das falhas encontradas.

2 AUDITORIA DE SISTEMAS DE INFORMAÇÃO

A auditoria aplicada a computação, segundo conceituação de Dias (2000), corresponde a atividade que engloba o monitoramento e a avaliação de sistemas, processos e operações

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

com o objetivo de verificar se as mesmas estão em conformidade com as políticas e objetivos, normas e procedimentos.

O trabalho de auditor está centrado em analisar o cenário atual e, em caso de não conformidade com as diretrizes estabelecidas, propor soluções para adequar o ambiente. A auditoria deve ser constante, visto que estamos tratando de um cenário que muda muito e de forma intensa.

Conforme Lento e Guimarães (2012), a partir do trabalho do auditor, é verificado se os processos estão sendo executados correta e constantemente. Além disso, pode ser verificado se os mesmos estão sendo executados de forma preventiva ou corretiva, como também se são independentes.

Para uma organização, pode ser adotado um padrão, que serve de base para tratar a maneira como a empresa se porta perante a informação. Esse padrão documentado recebe o nome de política de segurança da informação.

De acordo com Ferreira e Araújo (2006), a política de segurança pode ser compreendida como um conjunto de normas, métodos e procedimentos que visa a manutenção da segurança da informação. Tudo deve ser formalizado e divulgado entre os usuários. Nela, deve ser definido o escopo do que se quer proteger, visto que o ambiente que deve ser analisado vai além de hardware ou software, pois envolve pessoas e processos que devem ser levados em consideração.

Conforme descrito na NIC BR (2003), uma política pode ser entendida também como uma forma de atribuir responsabilidades e direito as pessoas que lidam com as informações. Sendo assim, são documentadas as atribuições que cada indivíduo tem em relação à segurança, sendo de conhecimento de todas as penalidades as quais está submetido quem estiver em desacordo com o estabelecido.

Uma das tarefas que são importantes para o bom funcionamento de todo o conjunto de tecnologia da informação diz respeito aos usuários. Cabe aos administradores da rede instruir os usuários a respeito da boa utilização dos recursos. Grande parte dos problemas decorre de falhas internas e, muitas das vezes são causados pelo despreparo e desconhecimento de conceitos básicos de segurança.

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

2.1 SEGURANÇA EM APLICAÇÕES WEB

Aplicações web são constantes alvos de ataques, onde invasores buscam aproveitar vulnerabilidades encontradas para o roubo de informações confidenciais. Seja no ambiente corporativo ou mesmo no âmbito pessoal, os ataques podem vir a gerar prejuízos não somente pelo vazamento de informações, mas também pelos danos potenciais de uma aplicação inoperante em um determinado período de tempo.

Estar vulnerável não somente significa a existência de uma falha de implementação ou erros na aplicação. O invasor pode se utilizar de algo concreto, como a relação estabelecida entre servidor e cliente para, a partir desse ponto, burlar a segurança.

A seguir, serão elencadas as vulnerabilidades mais comuns em aplicações web.

2.1.1 *Cross Site Scripting* ou XSS

Utiliza a relação de confiança entre o servidor da aplicação e o navegador para transporte de código malicioso, que geralmente é escrito em javascript, a fim de conseguir dados sensíveis, como o identificador da sessão do usuário.

Segundo OWSAP(2016), muitas aplicações permitem a postagem de conteúdo por parte de seus usuários. O XSS está ligado a esse conteúdo, que pode ser utilizado para solicitar informações ao usuário dentro da aplicação e o direcionar para fora dela. Como o navegador da vítima não consegue identificar o que corresponde ao trecho de código malicioso, este será executado assim que for detectado pelo navegador.

O ataque pode ser feito utilizando também os mecanismos de busca. Se a aplicação não possuir tratamento para tal, executará o código malicioso. Outra forma de fazer o ataque é inserindo no meio da página original aplicação, escrita em HTML, trechos de códigos a fim de se obter as informações desejadas. Para tornar o código ilegível e burlar mecanismos que se baseiam em tags para proteção, como por exemplo <script> ou <alert>, os códigos são escritos em hexadecimal.

Essa vulnerabilidade está ligada a falta de tratamento dos dados e conteúdo postado pelo usuário. A execução do código é imperceptível ao usuário infectado, já que é executada pelo navegador de forma transparente.

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

2.1.2 Injeção de SQL

Uma outra vulnerabilidade diz respeito a injeção de códigos com comando em *Structured Query Language* (SQL) para obter dados diretamente do banco. SQL corresponde a linguagem para gerenciamento dos dados utilizada pelos principais bancos de dados estruturados na modelagem relacional. Um estudo feito pela Owasp (2013) sobre vulnerabilidades em aplicações web apontou o *Structured Query Language Injection,* ou SQL *Injection* como a técnica mais utilizada no meio virtual para obtenção de dados de forma mal intencionada.

O ataque visa utilizar os dados de entrada do cliente no aplicativo para conseguir dados confidenciais, modificar, excluir ou atualizar registros. Conforme Owasp (2016), a técnica afeta um número grande de aplicações disponíveis atualmente, pois a arquitetura de grande parte se baseia em banco de dados SQL.

Em geral, a forma como as aplicações web constroem os comando SQL envolvem a sintaxe escrita pelos programadores com parâmetros informados pelos usuários. Dessa forma, há a possibilidade de o usuário explorar os parâmetros informados para obter dados do banco.

Conforme cita OWASP(2016), esses ataques ainda podem ser classificados em 3 diferentes classes:

- *Inband*: os dados são extraídos usando o mesmo canal que é usado para injetar o código;
- *Out-of-band*: os dados são recuperados em um outro canal, como por exemplo enviados para um e mail informado;
- Inferencial: não a transferência real dos dados, porém é possível reconstruir as informações através de solicitações particulares.

Como a exemplo do XSS, o SQL *Injection* consegue ser neutralizado se os formulários, campos de pesquisa, URLs, tiverem tratamento para os dados informados pelo usuário na aplicação.

2.1.3 Cross Site RequestForgery

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

Utiliza a relação de confiança entre o aplicativo e seu usuário legítimo. Geralmente fazendo uso de engenharia social, para explorar essa vulnerabilidade, o atacante necessita que a vítima esteja com uma sessão ativa na aplicação alvo. Nesse momento, o ataque pode ser concluído com sucesso caso o usuário receba um link ou acesse, no mesmo navegador, o aplicativo malicioso. Dessa forma, a aplicação maliciosa ou link inclui uma requisição ao aplicativo alvo, carregando os parâmetros necessário para a conclusão da transação.

Os aplicativos vulneráveis são aqueles em que as requisições são feitas de maneira estática, ou seja, sempre enviando os mesmos parâmetros para fazer a requisição.

Uma alternativa para tratar as requisições do CSRF é o *Synchronizer Token Pattern.* Diz respeito a enviar um token como um dos parâmetros da requisição. Com isso, tem-se a garantia de que a requisição está sendo feita, de forma que o token enviado seja comparado ao token armazenado na sessão do usuário, é gerado um novo token para armazenamento na sessão. Se o token for diferente, a requisição deve ser descartada pelo servidor.

2.1.4 Referência direta a objetos

Uma referência direta a um objeto expõe os dados de um sistema aos usuários. Seja um arquivo, diretório ou a chave de uma tabela, a referência direta permite que um usuário acesse dados aos quais não tem permissão. A fragilidade encontra-se na ideia de que o usuário sempre irá seguir os caminhos preestabelecidos pela aplicação.

Na própria aplicação são visualizados parâmetros que referenciam objetos internos. O usuário se aproveita dessa fragilidade para, alterando os parâmetros para ter acesso a outros dados da aplicação.

Como prevenção, deve-se verificar se o usuário tem permissão para acesso ao objeto ao qual está requisitando.

2.1.5 Broken Autenticação e Gestão de Sessões

O invasor utiliza-se de falhas da aplicação no gerenciamento de sessão ou na autenticação, como por exemplo, contas expostas, senhas, IDs de sessão, para representar um usuário legítimo.

De acordo com Owasp (2017) deve-se verificar as seguintes diretrizes:

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

- As credenciais de autenticação de usuário não são protegidas quando armazenadas usando hash ou criptografia.
- As credenciais podem ser adivinhadas ou substituídas por funções de gerenciamento de contas fracas (por exemplo, criação de contas, alteração de senha, recuperação de senha, IDs de sessão fracas).
- IDs de sessão são expostos na URL (por exemplo, reescrita de URL).
- IDs de sessão são vulneráveis a ataques de fixação de sessão .
- IDs de sessão não timeout, ou sessões de usuário ou tokens de autenticação, particularmente single sign-on (SSO) tokens, não são devidamente invalidados durante logout.
- Os IDs de sessão não são rodados após o login bem-sucedido.
- Senhas, IDs de sessão e outras credenciais são enviadas através de conexões não criptografadas.

Por exemplo, se alguém utiliza um computador público para acesso a uma determinada aplicação e, ao invés de fazer logout, apenas fecha o navegador, corre o risco de ter deixado a sessão ativa por um tempo, deixando aberta a possibilidade de que uma outra pessoa utilize-se dessa conta para obter dados.

2.2 FERRAMENTAS UTILIZADAS

Para auditoria das aplicações, foi utilizado o Kali Linux. Kali, conforme cita Pritchett e Smet (2013), é uma distribuição do Linux baseada no Debian e cuja finalidade é oferecer ferramentas para a utilização na auditoria e em testes de invasão, coleta de informações, identificação de vulnerabilidade, exploração, escalação de privilégios. Ao contrário da maioria das distribuições Linux, Kali é usado para fins de testes de penetração. Testes de penetração são uma maneira de avaliar a segurança de um sistema de computador ou rede, simulando um ataque.

A distribuição Kali possui uma série de ferramentas de análise, fornecendo uma avaliação abrangente e completa da estrutura de tecnologia da informação do ambiente auditado, através da qual podem ser elencados os itens que representam risco a segurança da informação, ponto em que o auditor deve oferecer alternativas para ajusta-los.

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

Utilizando esse sistema, a análise da estrutura será feita utilizando duas ferramentas: Whatweb e Vega.

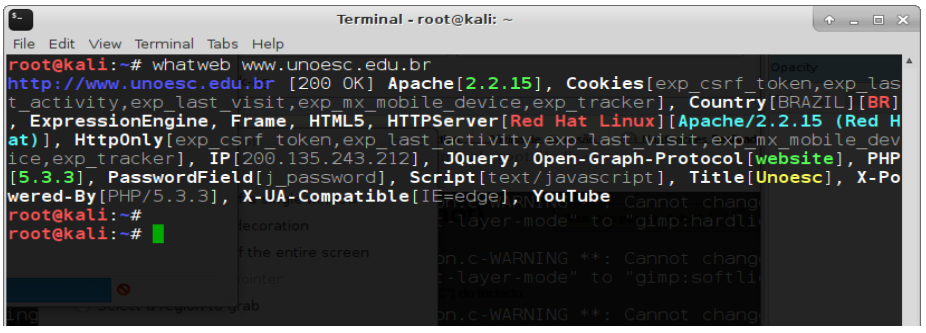
2.2.1 Whatweb

O Whatweb é uma ferramenta capaz de fazer uma análise e retornar quais são as tecnologias e detalhes da arquitetura da aplicação, endereço IP e dados referentes ao administrador do site, e-mails, erros de SQL, localização.

Embora a análise feita seja superficial, o Whatweb oferece ao auditor ou mesmo ao invasor uma visão global do ambiente, que pode servir como base para uma análise mais detalhada, focada nas tecnologias presentes. O potencial dessa ferramenta está associado ao fato de que ataques podem estar vinculados a uma tecnologia específica ou mesmo uma versão de um framework em que foi descoberta uma falha de segurança.

A figura 1 demonstra a tela do sistema Whatweb executando os testes para o site www.unoesc.edu.br:

Figura 1: Ambiente de execução da ferramenta Whatweb no site www.unoesc.edu.br

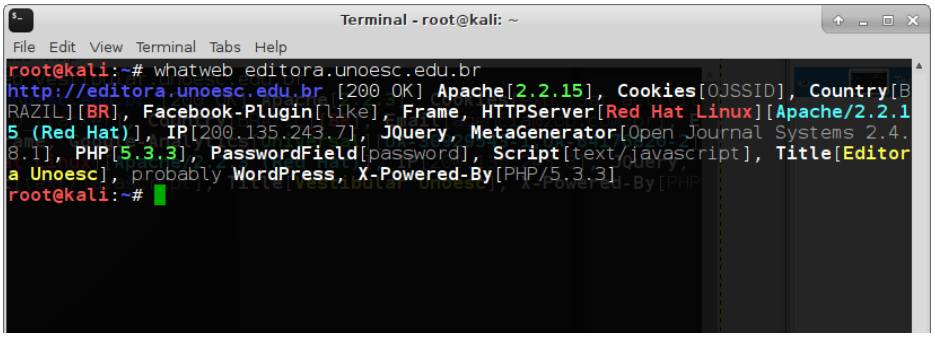


Fonte: O Autor

A figura 2 demonstra a tela do sistema Whatweb executando os testes para o site editora.unoesc.edu.br:

Figura 2: Ambiente de Execução da Ferramenta Whatweb para o site editora.unoesc.edu.br

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br



Fonte: O autor

Por meio da análise do resultado obtido com o Whatweb, temos embasamento para operacionalizar a execução de uma verificação mais abrangente com a utilização do Vega. Com a conclusão dessa primeira etapa da auditoria, pode ser observado que, apesar dos ambientes objetos desse estudo utilizarem gerenciador de conteúdo distintos, apresentam na construção de suas estruturas muitas ferramentas e tecnologias em comum.

2.2.2 Vega

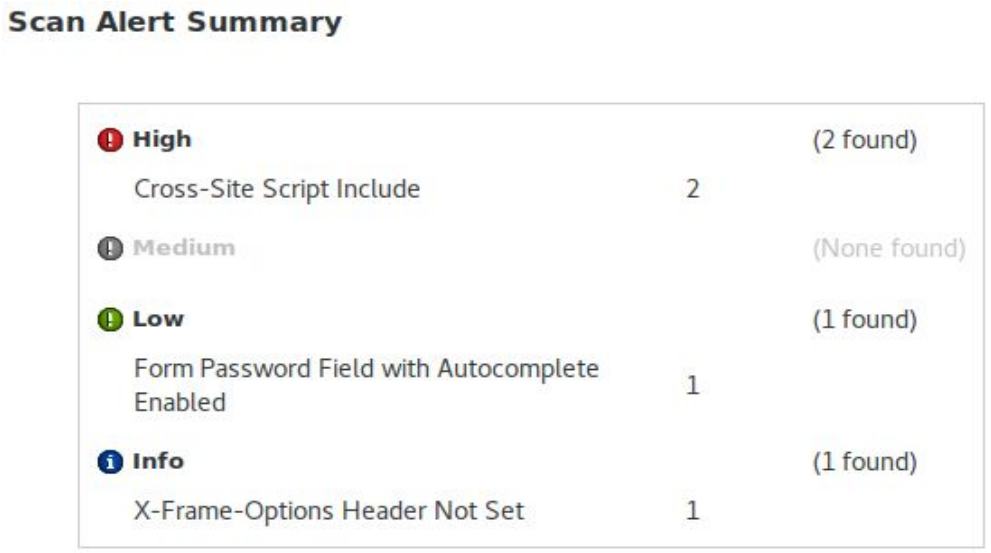
O Vega é uma ferramenta utilizada para realizar uma verificação mais detalhada das vulnerabilidades. Consegue encontrar e validar SQL *Injection*, XSS e outras vulnerabilidades. Após a análise concluída, são elencados possíveis soluções para os problemas relatados. Além disso, tudo o que aparentemente representa ameaça é classificado em diferentes níveis correspondendo ao risco para a aplicação.

Ao iniciar a análise, possibilita a inclusão de parâmetros e filtros, de acordo com o objetivo que se quer alcançar. Durante a execução, é possível acompanhar o andamento de acordo com os logs e alertas apresentados.

Para a análise dos ambientes objetos deste estudo, foram relacionadas todas as vulnerabilidades, permitindo um scanner completo de todos os aspectos. As figuras 3 e 4 representam o resultado desse scanner.

Figura 3: tela da ferramenta Vega mostrando os resultados no site www.unoesc.edu.br.

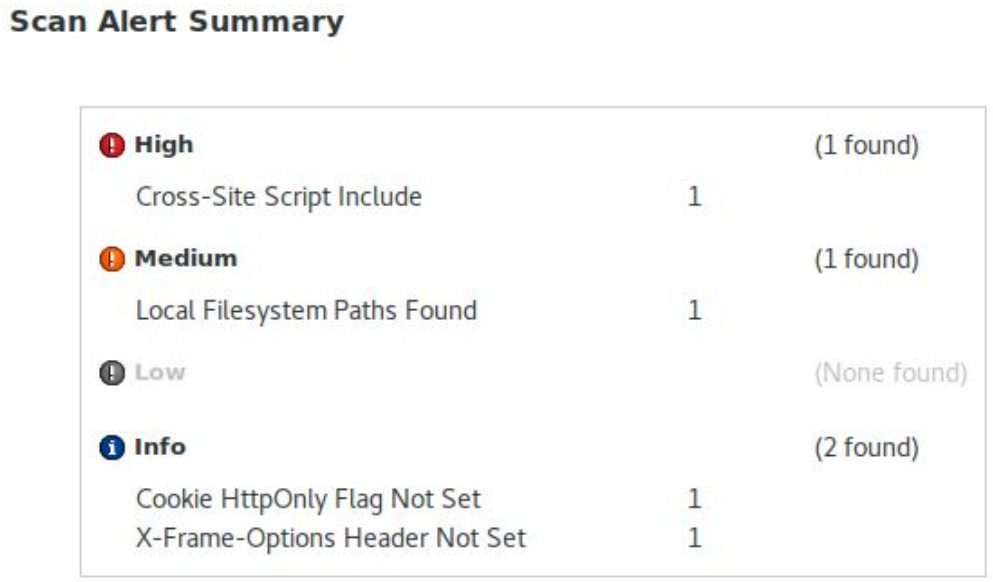
*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br



Fonte: O Autor

Figura 4: tela da ferramenta Vega mostrando os resultados para o site editora.unoesc.edu.br

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br



Fonte: O Autor

Em ambas as aplicações foi encontrada a possibilidade de explorar XSS, pois utilizam códigos javascript de fontes adversas ao da aplicação. Isso se configura como uma ameaça pois permite o acesso ao DOM por parte do servidor onde do código está hospedado, e se o mesmo ficar comprometido, refletirá nas aplicações que o utilizam.

Outra ameaça encontrada, porém de menor impacto, classificada nessa avaliação do Vega como média, diz respeito ao fato de a aplicação expor caminhos absolutos de arquivos. Através dessas informações, a aplicação revela como está estruturado seu sistema de arquivos, permitindo ao invasor explorar melhor os ataques, aumentando sua chance de sucesso.

Ainda foram identificadas outras vulnerabilidades que, apesar de não apresentarem grande risco para a segurança, também devem ser analisadas. Classificada como risco baixo, o scanner identificou a existência de um formulário com atributo de preenchimento automático não foi definido como desativado.Apresenta risco na medida em que o navegador local armazena as credenciais de um usuário localmente, tornando possível a recuperação por outro usuário.

Identificadas apenas como informação pelo Vega, foi identificado um cookie sem a propriedade HttpOnly. É uma medida de segurança que ajuda a mitigar o risco mediante um ataque que visa os cookies de sessão dos usuários.

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

3 CONCLUSÃO

Aplicações web são potenciais alvos, sendo que o atacante pode utilizar de diferentes caminhos e ferramentas para explorar falhas.

O uso de ferramentas para testes de invasão, como o Vega e Whatweb permitem que sejam descobertas e solucionadas vulnerabilidades, minimizando os riscos. Durante os testes, através da simulação de um ataque, é possível mensurar o impacto gerado sobre a aplicação, elencando itens classificados de acordo com o possível dano, que serve de apoio para a estruturar as correções necessárias.Tendo em vista que os ambientes estão em constante mudança, seja pela implementação de novas funcionalidades, ou mudanças na arquitetura, a auditoria deve ser feita periodicamente.

Em decorrência da utilização de frameworks em sua estrutura, as aplicações analisadas já possuem soluções implementadas para muitas vulnerabilidade. Além de possuir configurações de segurança, são liberadas novas *releases* do framework, como respostas a falhas encontradas.

O Estudo conclui que o ambiente WEB dos sites analisados no domínio da Unoesc, unoesc.edu.br, estão em conformidade e já atualizados quanto a prevenção dos principais problemas de segurança da atualidade, sugerindo que para uma segurança mais efetiva, devam ser submetidos a uma quantidade maior de testes e avaliações por ferramentas não contempladas no escopo deste estudo, assim como a verificação do ambiente utilizando *checklists* voltados a obtenção de uma certificação de segurança de informação (ISO/NBR).

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br

REFERÊNCIAS

DIAS, Claudia. Segurança e auditoria da tecnologia da informação. Rio deJaneiro: Axcel Books do Brasil, 2000. 218 p.

FERREIRA, Fernando Nicolau Freitas; ARAÚJO, Márcio Tadeu de. **Política de Segurança da Informação**: guia prático para elaboração e implementação. 2. ed. Rio de Janeiro: Ciência Moderna. 2006.

FONTES, Eduardo. Segurança da informação: o usuário faz a diferença. –São Paulo: Saraiva, 2006.

LENTO, Luiz Otávio Botelho; GUIMARÃES, Márcio Guisi. Auditoria desegurança da informação: livro virtual. Palhoça: Unisul Virtual, 2012. 116 p.

NIC BR Security Office (NBSO). **Práticas de Segurança para Administradores de Redes Internet** Disponível em: http://www.cert.br/docs/seg-adm-redes/seg-adm-redes.html. Acesso em: maio de 2016. Versão 1.2. 16 de maio de 2003.

OWASP, **Testing for SQL Injection**. Disponível em: https://www.owasp.org/index.php/Testing_for_SQL_Injection_(OTG-INPVAL-005). Acesso em 10 de abril de 2017.

OWSAP, **Top 10 2013-Top 10.** Disponível em: https://www.owasp.org/index.php/Top_10_2013-Top_10. Acesso em: 10 de abril de 2017.

OWSAP, **SQL Injection.** Disponível em: https://www.owasp.org/index.php/SQL_Injection, Acesso em 10 de abril de 2017.

PRITCHETT, Willie. L.; SMET, David De. (2013). Kali Linux CookBook. Packt Publishing Ltd., Birmingham.

RORRATO, Rodrigo; DIAS, Evandro Dotto. Segurança da informação deprodução e operações: um estudo sobre trilhas de auditoria em sistemas debanco de dados. JISTEM - Journal of Information Systems andTechnology Management. vol. 11, no. 3, sept/dec., 2014 pp. 717-734. doi:10.4301/S1807-17752014000300010

*Pós-graduado em gestão de tecnologia da informação pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: alan.bernardi@unoesc.edu.br
** Mestre em ciência e biotecnologia pela Universidade do Oeste de Santa Catarina – UNOESC; e-mail: fabiano.wonzoski@unoesc.edu.br