0CENTRO UNIVERSITÁRIO UNIEVANGÉLICA
CAMPUS ANÁPOLIS
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

JEAN CARLOS GOMES MARTINS
VANESSA MOTA ALVES

PROCESSO DE GESTÃO DE SEGURANÇA DA INFORMAÇÃO: DEFINIÇÃO E APLICAÇÃO EM UM SISTEMA LABORATORIAL

Anápolis - GO
2017 - 02

JEAN CARLOS GOMES MARTINS
VANESSA MOTA ALVES

PROCESSO DE GESTÃO DE SEGURANÇA DA INFORMAÇÃO: DEFINIÇÃO E APLICAÇÃO EM UM SISTEMA LABORATORIAL

Monografia apresentado ao Curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA - como requisito parcial à aprovação na disciplina Trabalho de Conclusão de Curso II sob orientação da Profª. Ma. Renata Dutra Braga.

Anápolis - GO
2017 - 02

JEAN CARLOS GOMES MARTINS
VANESSA MOTA ALVES

PROCESSO DE GESTÃO DE SEGURANÇA DA INFORMAÇÃO: DEFINIÇÃO E APLICAÇÃO EM UM SISTEMA LABORATORIAL

Monografia apresentado ao Curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis - UniEVANGÉLICA - como requisito parcial à aprovação na disciplina Trabalho de Conclusão de Curso II sob orientação da Profª. Ma. Renata Dutra Braga.

Aprovada em ( ) de ( ) de 2017

BANCA EXAMINADORA

_____

Profª. Ma. Renata Dutra Braga- Orientadora

_____

Convidado

_____

Convidado

AGRADECIMENTOS

A Deus por nos proporcionar saúde e sabedoria.

A nossa família e amigos que nos ampararam no decorrer deste trabalho.

À Instituição por nos fornecer a infraestrutura e o conhecimento necessário para atingirmos nossos objetivos acadêmicos.

À Profª. Ma. Renata Dutra Braga por compartilhar seu valioso conhecimento na última e árdua fase do projeto.

À Profª. Dra. Kátia Kelvis Cassiano Lozano pelo apoio no planejamento e elaboração deste trabalho, exercendo seu papel com a maior boa vontade possível e que, com toda certeza, foi fundamental para o resultado final.

“O que prevemos raramente ocorre; o que menos esperamos geralmente acontece. ”

(Benjamin Disraeli)

RESUMO

Contextualização: O trabalho aborda a segurança da informação em sistemas *web*, tendo como base o estudo do ambiente do Sistema Laboratorial de Análises Clínicas (SisLAC), de propriedade da Fábrica Turing de Tecnologias (FTT) do Centro Universitário de Anápolis UniEVANGÉLICA. **Objetivo:** Definir um processo de gestão de segurança da informação com vistas à melhoria contínua dos processos de desenvolvimento de software. **Metodologia:** Análises das vulnerabilidades técnicas do sistema e classificação das mesmas foram realizadas, segundo prerrogativas de normas técnicas de referência no âmbito da segurança da informação (ISO 27001). **Resultados:** Um laboratório de teste foi definido e os resultados obtidos foram utilizados como referência para a formulação de um processo de gestão de segurança da informação. Além disto, níveis de riscos foram apurados com base nas vulnerabilidades identificadas. **Conclusão:** Uma visão estratégica para a segurança da informação na FTT foi definida, de forma que os papeis e responsabilidades exerçam suas atividades durante todo do processo de desenvolvimento do software, aplicando as boas práticas destacadas por normas de referência internacional.

Palavras-chave: Segurança da informação. Vulnerabilidades em aplicações *web*. Melhoria Contínua. Análise de riscos.

ABSTRACT

Contextualization: The work addresses an information security in web systems, based on the study of the environment of the System of Clinical Analysis (SisLAC), owned by the Turing Factory of Technologies (TFT) of UniEVANGÉLICA University Center. **Objective:** Define an information security management process with perspectives of continuous production of software development processes. **Methodology:** Analyzes of the technical vulnerabilities of the system and the qualification of the editions were carried out according to the prerogatives of reference technical standards in the field of information security (ISO 27001). **Results:** A test laboratory was defined and the results obtained as applications for a formulation of an information security management process. In addition, risk levels are calculated based on identified vulnerabilities. **Conclusion:** A strategic vision for information security in the TFT has been defined, so that roles and responsibilities carry on throughout the software development process, applying good practices highlighted by international benchmarks.

Keywords: Information security. Vulnerabilities in web applications. Continuous Improvement. Risk Analysis.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURA E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
BPMN	*Business Process Modeling Notation*
Cert.br	Centro de Estudos, Respostas e Tratamento de Incidentes de Segurança no Brasil
FTT	Fábrica de Tecnologias Turing
HTTP	*HyperText Transfer Protocol*
IEC	*International Electrotechnical Commission*
IP	*Internet Protocol*
ISO	*International Organization for Standardization*
LDAP	*Lightwight Directory Access Protocol*
Nmap	*Network Mapper*
NoSQL	*Not Only Structured Query Language*
OWASP	*Open Web Application Security Project*
PDCA	*Plan, Do,Check, Act*
SisLAC	Sistema Laboratorial de Análises Clínicas
SQL	*Structured Query Language*
TDI	Teste de Invasão
URL	*Uniform Resource Locator*
XSS	*Cross-site Scripting*
ZAP	*Zed Attack Proxy*
Zenmap	*Zen Mapper*

SUMÁRIO

1 INTRODUÇÃO

Na rede mundial de computadores existem mais de um bilhão de *websites* ativos (NETCRAFT, 2017). O grau de importância desses *websites/*aplicações para as organizações cresce, significativamente, a cada ano, uma vez que constituem oportunidades para que as mesmas atinjam seus objetivos de negócio. Entretanto, em média 70% dos *websites* têm vulnerabilidades que comprometem a segurança dos dados do sistema, que podem vir a ser exploradas por indivíduos mal-intencionados (ACUNETIX, 2017).

Existem inúmeros fatores que motivam um indivíduo a realizar ações maliciosas em sistemas de informação. As principais são: *motivação financeira*, que visa obter recursos financeiros através de golpes; *motivação ideológica*, que visa divulgar suas crenças de maneira forçada; *motivação comercial*, causando prejuízos a corporações vítimas e *motivação de demonstração de poder*, com propósito de obter alguma vantagem pessoal no futuro (CERT.BR, 2012).

Negligenciar práticas de segurança da informação pode gerar problemas graves à privacidade dos dados do sistema, além de causar impacto negativo à qualidade do software, como, por exemplo, não conformidade com normas técnicas, baixa confiabilidade do sistema, aumento de custo, dentre outros.

No âmbito da segurança da informação, existem alguns padrões de gerenciamento com vistas as melhores práticas. Estes padrões foram publicados por organizações internacionais (*International Organization Standardization* – ISO e *International Electrotechnical Comission* – IEC) e servem como referência na adoção de processos de gestão de segurança da informação. Este trabalho utiliza, primariamente, como referência a ISO/IEC 27001 (Sistema de gerenciamento de segurança da informação) e a ISO 27002 (Boas práticas em gestão de segurança da informação).

Uma forma de se resguardar é realizar análises críticas de segurança que permitam modelar a situação do sistema no que tange à gestão de vulnerabilidades, e tomar medidas contra os riscos identificados, assim como sugere a norma ISO/IEC 27001:

> Informações sobre vulnerabilidades técnicas dos sistemas de informação em uso devem ser obtidas em tempo hábil, com a exposição da organização a estas vulnerabilidades avaliadas e tomadas as medidas apropriadas para lidar com os riscos associados (2013, p. 24).

Os danos que a exploração de uma vulnerabilidade pode causar são relativos e dependentes da natureza do negócio. A magnitude desse impacto, geralmente, está relacionada

à quantidade de usuários ativos ou à quantidade de movimentações financeiras que o sistema controla.

O Sistema Laboratorial de Análises Clínicas (SisLAC), sistema desenvolvido e mantido pela Fábrica de Tecnologias Turing (FTT), foi escolhido como ambiente de estudo de caso. A Motivação para escolha desse sistema surgiu da necessidade identificada através dos responsáveis pelo desenvolvimento e manutenção do mesmo, os quais relataram que não há uma equipe específica para garantir que a segurança da informação seja parte integrante de todo o ciclo de vida dos sistemas mantidos pela FTT, incluindo o SisLAC, assim como nenhuma ação preventiva de segurança é realizada nos sistemas já implantados.

Como base neste cenário, a seguinte pergunta de pesquisa norteou o desenvolvimento deste trabalho: que conjunto de atividades pode ser aplicado para promover a gestão da segurança da informação em um sistema laboratorial de análises clínicas (SisLAC)?.

O objetivo geral deste estudo é propor um processo de gestão da segurança da informação, servindo de guia para o planejamento, execução e monitoramento da segurança da informação em sistemas Web.

Para alcançar o objetivo proposto, as seguintes atividades foram desenvolvidas: (a) estudo sobre ferramentas automatizadas para identificação de vulnerabilidades Web, ferramentas estas que foram capazes de encontrar diversas vulnerabilidades no ambiente de estudo, (b) identificação das vulnerabilidades do sistema, classificando-as qualitativamente, (c) Prover subsídios para a implementação da gestão da segurança da informação no SisLAC, através de um planejamento de respostas aos riscos associados, embasadas em normas internacionais de melhores práticas em segurança da informação.

O texto está organizado em cinco capítulos. No segundo, a fundamentação teórica, base para a realização de todo o projeto, é descrita. O terceiro capítulo apresenta a estrutura do estudo de caso proposto, ao passo que o quarto descreve os resultados obtidos. Por fim, no quinto capítulo as considerações finais são apresentadas.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, será abordado o arcabouço teórico pertinente a este trabalho. Inicialmente serão apresentados conceitos de segurança da informação e, em seguida, teste de invasão e vulnerabilidades, com ênfase em aplicações *web*.

2.1 SEGURANÇA DA INFORMAÇÃO

Com a popularização da internet e dos recursos computacionais, tem-se maior demanda pela proteção da informação – ativo valioso das organizações. A segurança da informação tem o objetivo de proteger a informação e seus sistemas de toda a gama de incidentes que possam causar impacto ao negócio.

A norma ABNT ISO/IEC 27000 (2014) descreve a segurança da informação como sendo a responsável por envolver a aplicação e a gestão de medidas de segurança adequadas para um amplo conjunto de ameaças, objetivando assegurar o sucesso e a continuidade do negócio por meio da minimização dos impactos de incidentes de segurança da informação. Ainda segundo a norma, com intuito de proteger a informação e maximizar o retorno sobre o investimento, a organização deve prezar pela tríade da segurança da informação: confidencialidade, integridade e disponibilidade.

A confidencialidade é a garantia de que o acesso à informação seja obtido apenas por aqueles que têm autorização. A integridade é a garantia de que a informação é fidedigna e que não sofreu alterações por pessoas não autorizadas. Por fim, a disponibilidade trata da garantia de que a informação estará disponível aos usuários autorizados sempre que necessário.

2.2 SEGURANÇA PROATIVA VERSUS SEGURANÇA REATIVA

Segundo o controle A.6.1.5 da norma ABNT ISO/IEC 27001 (2013): “Segurança da informação deve ser considerada em todo o ciclo de vida do gerenciamento de projetos, independentemente do tipo do projeto”. As organizações, em geral, aderem a esse controle, porém na maioria das vezes apenas soluções reativas são modeladas, partindo do pressuposto de que não existem ameaças imediatas. Dessa forma, não previnem o incidente atuando apenas quando incidentes ocorrem. Por controle entende-se como uma “medida que está modificando o risco” (ABNT ISO/IEC 27000, 2013, p. 3).

Planejar respostas rápidas aos incidentes é importante quando se trata da proteção aos ativos de tecnologia da informação. Porém recomenda-se, sobretudo, que tais medidas sejam alinhadas a uma estratégia de prevenção, que possa detectar ameaças a priori, assim como possíveis falhas nos sistemas, encontrar suas causas raízes e adotar medidas para solucioná-las.

Vale ressaltar que uma estratégia puramente reativa é arriscada do ponto de vista da segurança, pois ainda que exista uma resposta rápida a um determinado incidente, efeitos irreversíveis ao negócio podem ser gerados pelo mesmo.

Dada a importância da prática proativa, é imprescindível que as organizações implementem processos que permitam identificar suas vulnerabilidades e fraquezas.

> Diante de novas tendências, táticas de invasões e da rapidez com que se move o mundo digital, é de suma importância que administradores de redes ou sistemas tenham em mente que "*blackhats*[1]" estão em constante evolução e são inúmeros os métodos utilizados nas práticas de invasões. Conhecendo tal ameaça, é fundamental que os profissionais envolvidos neste meio estejam treinados e preparados para lidar com situações que envolvam a segurança da informação a fim de manter um ambiente seguro e proteger os dados que ali estão inseridos (GIAVAROTO; SANTOS, 2013, p. 4).

2.3 TESTE DE INVASÃO

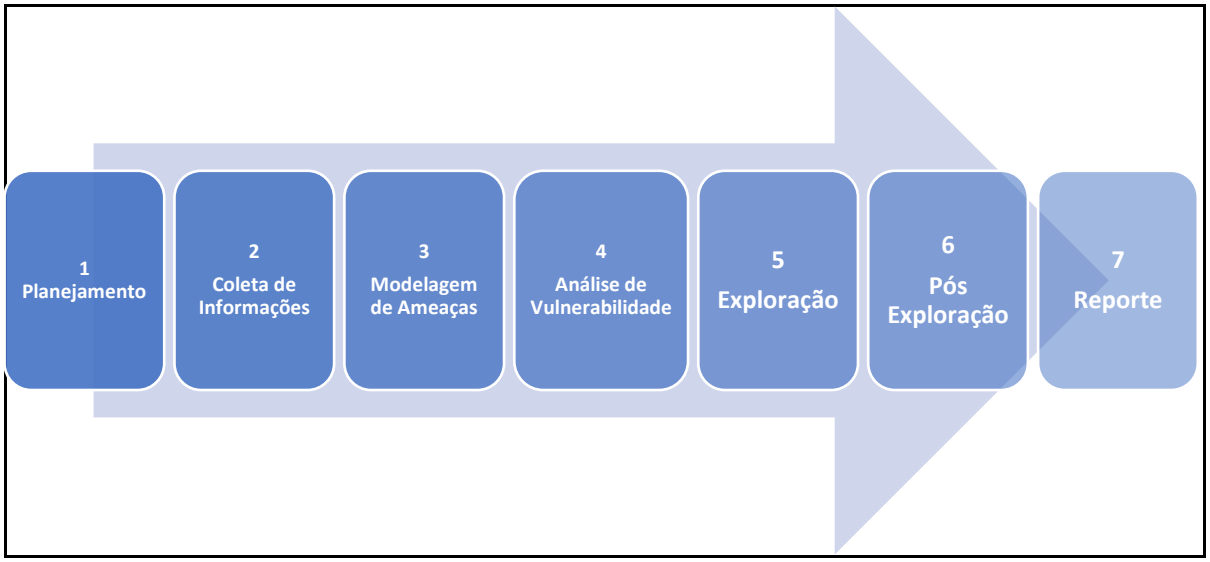
Teste de Invasão (TDI), também conhecido como Teste de Intrusão ou Teste de Penetração, é definido como um método de atacar vulnerabilidades do sistema de forma semelhante à atuação dos ataques maliciosos reais (MUNIZ; LAKHANI, 2013).

O objetivo principal dos testes de invasão é determinar as fraquezas da segurança de redes ou aplicações *web*. Os testes podem ser feitos com ferramentas automatizadas ou manualmente e o escopo do teste varia de acordo com os objetivos do negócio.

A Figura 1 apresenta as fases do teste de invasão, sendo elas: Planejamento, Coleta de informações, Modelagem das ameaças, Análise de vulnerabilidades, Exploração, Pós Exploração e Relatório (WEIDMAN, 2014).

[1] Indivíduo que usa seu conhecimento avançado em sistemas e redes de computadores para fins ilícitos, pela mídia é conhecido como Hacker (GIAVAROTO; SANTOS, 2013, p. 2).

Figura 1 – Fases do Teste de Invasão



Fonte: Elaborado pelos autores.

A primeira fase é a de planejamento (Figura 1). Nela, a equipe responsável pelo teste busca se comunicar com as partes envolvidas a fim de alinhar suas expectativas. É o momento de entender quais serão os ativos englobados nesse teste, e qual o grau de importância desses ativos para o negócio. O critério de sucesso e a criticidade dos ativos envolvidos também são discutidos nessa fase. Há alguns procedimentos importantes nessa fase, são eles: definir escopo, definir a janela de testes e obter termo de autorização (WEIDMAN, 2014).

Definir o escopo é de suma importância para o sucesso do processo como um todo. Nesse momento o foco é definir a amplitude do teste, o que será testado e suas limitações. É necessário, também, acordar com as partes envolvidas os períodos em que os testes poderão ser realizados, ou seja, a sua janela de realização. Ainda na fase de planejamento, é recomendável que seja feito um termo de autorização formal, já que a execução de um teste de invasão é um processo delicado.

A segunda fase do teste é a de coleta de informações. Nessa fase é realizado um apanhado de dados sobre o alvo do teste. São utilizadas ferramentas para identificar informações como: portas abertas de protocolos de redes, dispositivos que estão em uso e informações diversas sobre arquitetura da rede e de servidores (WEIDMAN, 2014).

A terceira fase é a modelagem de ameaças, na qual o testador utiliza as informações coletadas na fase anterior para determinar o impacto da ameaça para o cliente. Esta avaliação permite desenvolver um plano de ação e um método de ataque (WEIDMAN, 2014).

A análise de vulnerabilidades é a quarta fase. Durante esta etapa, as primeiras

vulnerabilidades são encontradas. Para isso, quase sempre são utilizadas ferramentas automatizadas que possuam um banco de dados de vulnerabilidades. Porém, por mais que as ferramentas sejam eficientes, o pensamento crítico em testes manuais não é dispensável. Essa etapa é requisito obrigatório para se obter conformidade com a norma ISO/IEC 27001.

A quinta fase é a parte do teste em que as vulnerabilidades encontradas na fase anterior são postas a prova, evitando, assim, falsos positivos. É, por consequência, a fase do teste que requer maior empenho da equipe. "O sucesso desta etapa é fortemente dependente de esforços anteriores. A maioria dos *exploits*[2] são desenvolvidos para vulnerabilidades específicas e pode causar consequências indesejáveis " (M1UNIZ; LAKHANI, 2013, p. 19). São objetivos dessa fase: explorar vulnerabilidades, capturar dados não autorizados, identificar documentos, atacar outros sistemas ou aplicações, executar engenharia social, dentre outros (MUNIZ; LAKHANI, 2013).

A sexta fase é a pós-exploração. A partir das informações recolhidas da fase de exploração, a equipe deve realizar uma série de atividade como escalar os privilégios, manter o acesso e até mesmo utilizar um alvo explorado para atacar outro.

A fase final do teste é a elaboração do relatório, documentando informações relevantes para a organização, incluindo as vulnerabilidades encontradas, o nível de insegurança, a priorização dos riscos e um guia para tratamento desses riscos. Essa documentação é fundamental, segundo a norma ABNT ISO/IEC 27001 (2013, p. 10): "a organização deve reter informação documentada dos resultados das avaliações de risco de segurança da informação".

2.3.1 Tipos de Testes de Invasão

Há diversas maneiras de executar um teste de invasão. Os principais são: caixa branca, caixa preta e caixa cinza (MUNIZ; LAKHANI, 2013).

Caixa branca é quando o testador tem conhecimento apurado sobre o sistema em questão. Os objetivos do teste são claramente definidos e os resultados do teste são normalmente esperado. É focado em um objetivo de negócio específico, como análise de aplicações *web*, em que se pode analisar abertamente o código-fonte e o servidor em busca de vulnerabilidades de segurança. Esse tipo de teste tende a necessitar de menos esforço, visto que a coleta de informações é reduzida (MUNIZ; LAKHANI, 2013).

[2] Explorar, obter vantagem de uma falha (MUNIZ; LAKHANI, 2013).

Caixa preta é quando o testador não possui informações do alvo. Todas as informações são obtidas a partir de ferramentas e técnicas manuais, o que gera um custo mais alto a esse tipo de teste, visto que a busca por informações consumirá mais tempo (MUNIZ; LAKHANI, 2013).

Caixa cinza é a junção de características dos testes de caixa branca e preta. Esse tipo de teste garante ao testador o conhecimento de algumas informações de seu alvo. Essa abordagem é muito utilizada visto que é a mais próxima do mundo real, já que os agentes de ataque reais não buscam alvos aleatórios, eles atacam alvos determinados e, supostamente, já possuem algumas informações características dos mesmos (MUNIZ; LAKHANI, 2013).

2.4 VULNERABILIDADES

2.4.1 Conceitos Básicos

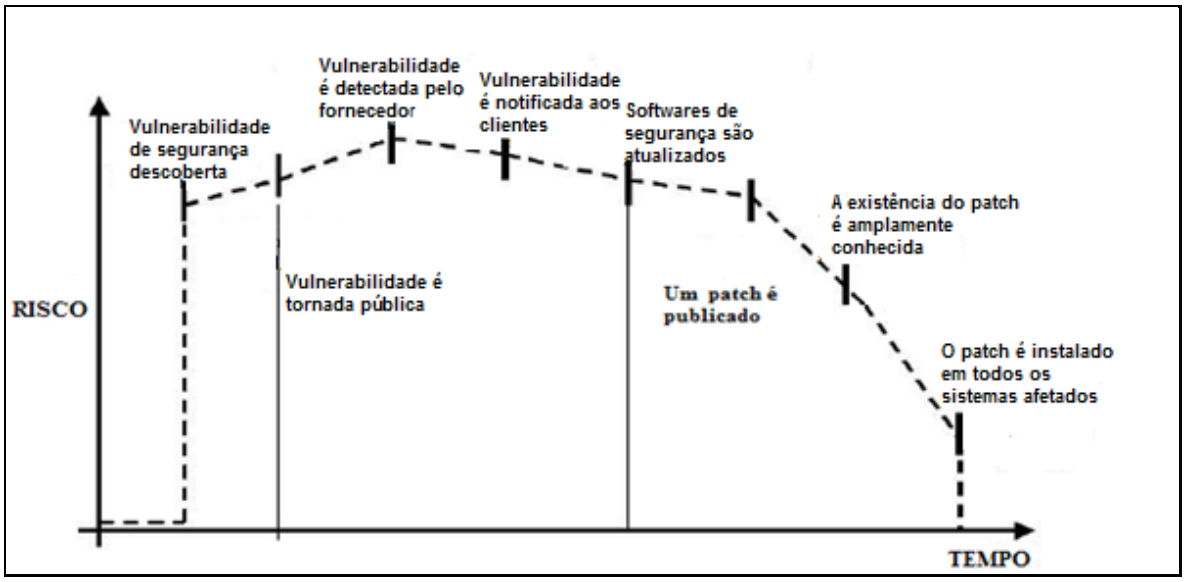
O Centro de Estudos, Respostas e Tratamento de Incidentes de Segurança no Brasil (CERT.BR, 2012), em sua cartilha de segurança para internet, define vulnerabilidade como uma condição, que pode resultar em uma violação de segurança se explorada intencionalmente ou não por um agente de ataque. Ela pode ser entendida como uma falha, brecha ou rachadura de um ativo. A ABNT ISO/IEC 27005 (2011), que trata da gestão de segurança da informação, cita alguns cenários onde é possível identificar vulnerabilidades: software, ambiente físico, *hardware*, recursos humanos e rotinas de gestão.

Entretanto, a existência de uma vulnerabilidade não significa necessariamente prejuízo, pois é necessário que haja uma ameaça que possa utilizar tal vulnerabilidade para causar dano a um ativo. Embora uma vulnerabilidade que seja atualmente desprovida de ameaças não necessite de implementação de controles imediatos, é razoável que seja identificada como tal e, portanto, monitorada (ISO/IEC 27005, 2011).

2.4.2 Janela de Exposição

Novas vulnerabilidades são descobertas constantemente. Da sua descoberta até seu declínio, há diferentes níveis de exposição e, consequentemente, diferentes níveis de risco. A Figura 2 mostra a variação desses níveis em relação ao tempo de vida das vulnerabilidades.

Figura 2 – Janela de Exposição



Fonte: Adaptado de OWASP (2008, p.17).

Inicialmente, quando descoberta, uma vulnerabilidade possui alto risco, visto que não há nenhuma medida que a impeça de ser explorada, pois o seu conhecimento ainda não foi publicamente divulgado. Esse evento também é conhecido como “dia zero” (MUNIZ; LAKHANI, 2013, p. 236).

Em um segundo momento, a vulnerabilidade é tornada pública e seu risco tende a aumentar consideravelmente. Entretanto quando a sua existência é divulgada aos usuários finais, por meio dos fornecedores do sistema, os riscos começam a regredir.

Em seguida, os riscos são gradativamente reduzidos à medida que softwares de segurança são atualizados e principalmente pela aplicação de *patchs*[3] contendo soluções para tal vulnerabilidade.

2.4.3 Vulnerabilidades em Aplicações *Web*

Existem inúmeros tipos de vulnerabilidades em aplicações *web*. A Organização Internacional de Segurança em Aplicações *Web* (OWASP), periodicamente produz artigos sobre os dez riscos de segurança mais críticos em aplicações *web*. Segundo a OWASP (2017b), em primeiro lugar está a vulnerabilidade de Injeção, seguido de Quebra de Autenticação e

[3] Pacotes com atualizações de software que, geralmente, contêm correções de vulnerabilidades (MUNIZ; LAKHANI, 2013).

Gerenciamento de Sessão e *Cross-Site Scripting* (XSS). Essas vulnerabilidades citadas serão detalhadas a seguir, uma vez que são as mais críticas da atualidade em aplicações *Web*. Contudo, destaca-se que a escolha das vulnerabilidades que serão tratadas será com base no escopo do ambiente alvo da análise.

2.4.3.1 Injeção

As falhas de injeção, como injeção de SQL ou de sistema operacional, ocorrem quando dados não legítimos são enviados para o interpretador para a realização de uma consulta, de forma a adulterar seus resultados e conseguir acesso não autorizado (OWASP, 2017b).

Essa vulnerabilidade tornou-se bastante perigosa por quatro motivos: fácil exploração, prevalência comum, detecção média e impactos severos. É considerada de fácil exploração, porque basta o agente de ataque enviar ações em forma de texto simples, de forma a enganar o interpretador utilizando uma sintaxe não convencional (OWASP, 2017b).

A prevalência é comum, pois este tipo de injeção, seja via consultas SQL(*Structured Query Language*), NoSQL (*Not Only Structured Qery Language*) ou LDAP (*Lightwight Directory Acess Protocol*) é encontrada frequentemente, particularmente em códigos legados. Isso faz com que os agentes de ataque busquem encontrar essa vulnerabilidade como primeiro recurso.

A detecção é média, pois são difíceis de serem descobertas através de testes, mas são fáceis de serem detectadas através de análise de código.

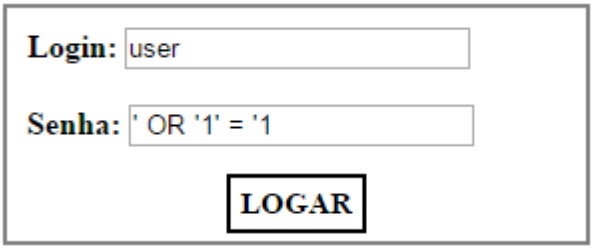
O ponto principal dessa vulnerabilidade é seu impacto severo. As consequências de um ataque de injeção podem ser devastadoras, afetando diretamente a confidencialidade e integridade, incluindo perda e corrupção de dados, negação de acesso e, em casos mais graves, o comprometimento completo do servidor.

2.4.3.2 Injeção de SQL

A Injeção de SQL (SQL *Injection*) pode ser executada de forma a provocar um erro na aplicação para que, em resposta, alguma informação importante seja recebida pelo agente de ataque.

A Figura 3 ilustra a tentativa de um ataque de injeção de SQL na autenticação de usuário. A utilização dessa sintaxe pode fazer com que, internamente, a instrução que verifica a existência do *login* e senha digitados seja manipulada. Em sistemas convencionais, a consulta seria, por exemplo: SELECT * FROM usuarios WHERE login = ‘usuario_login’ AND senha = ‘usuario_senha’.

Figura 3 – Exemplo de Injeção de SQL na autenticação de usuário



Fonte: Elaborado pelos autores.

Utilizando os dados de entrada da Figura 3, a consulta SQL seria: SELECT * FROM usuarios WHERE *login* = ‘*user*’ AND senha = ‘’OR ‘1’ = ‘1’. Dessa forma, independente do *login* e senha serem válidos, a condição será sempre verdadeira, permitindo, assim, acesso não autorizado ao sistema.

2.4.3.3 Quebra de Autenticação e Gerenciamento de Sessão

As falhas de quebra de autenticação e gerenciamento de sessão estão relacionadas à má implementação das funções de autenticação, permitindo que agentes de ataque comprometam senhas, chaves e *tokens*[4] de sessão, ou explorem a falha para assumir controle da identidade de outros usuários (OWASP, 2017b). É considerada uma falha problemática, que possui as seguintes características: dificuldade de exploração média, prevalência generalizada, detecção média e impactos severos.

Para obter êxito na exploração dessa vulnerabilidade, o agente de ataque necessita apenas utilizar de falhas nas funções de autenticação ou gerenciamento de sessão, sendo considerada média sua dificuldade de exploração (OWASP, 2017b).

Possui prevalência generalizada, já que essa falha é frequentemente encontrada em sistemas *web*, visto que a implementação correta das funções de autenticação e gerenciamento

[4] Conjunto de símbolos utilizados para validar uma sessão (OWASP, 2017c).

de sessão são mais trabalhosas, e por esse motivo é mais trabalhosa e por isso é muitas vezes negligenciada.

A detecção é considerada média, pois apesar de ser uma tarefa complexa, geralmente essas falhas são encontradas nas áreas de *logout*, gestão de senhas e tempo de expiração.

O impacto que esse tipo de vulnerabilidade pode causar é severo, uma vez que a confidencialidade, autenticidade e integridade serão infringidas nos ataques bem-sucedidos. Danos maiores ao sistema serão verificados a partir da supressão de identidade de contas com privilégio avançado (OWASP, 2017b).

2.4.3.4 Cross-Site Scripting (XSS)

O *Cross-Site Scripting* (XSS) é uma falha que permite ao agente de ataque enviar dados não-confiáveis para o sistema, pois as entradas do sistema não estão apropriadamente filtradas ou validadas. A dificuldade de exploração dessa vulnerabilidade é considerada média, sua prevalência é muito difundida e de média detecção, podendo gerar impactos de nível moderado (OWASP, 2017b).

A exploração média justifica-se pelas diversas fontes de dados que podem servir como vetor de ataque. Trata-se de uma falha bastante difundida em aplicações *web*. A detecção média também é fator preocupante dessa vulnerabilidade, já que não é tão complexa sua detecção através de testes automatizados ou em análise de código (OWASP, 2017b).

Seu impacto é mais brando que as vulnerabilidades citadas anteriormente. Contudo, a confidencialidade, integridade e autenticidade podem ser afetadas pela exploração dessa vulnerabilidade. Agentes de ataque, através da execução de *scripts* no navegador da vítima, podem sequestrar sessões, inserir conteúdo, desfigurar a página, dentre outros (OWASP, 2017b).

3 ESTUDO DE CASO: DESENHO METODOLÓGICO

3.1 DESENHO METODOLÓGICO

Quanto a natureza do trabalho, caracteriza-se como uma pesquisa aplicada, com abordagem qualitativa. Para GOLDENBERG (2009 apud, GERHARDT e SILVEIRA, 1997), a pesquisa qualitativa não tem como objetivo chegar a números exatos como resultados, mas indicar o caminho para tomada de decisão correta sobre um determinado problema. Para alcançar os objetivos propostos, foi empregado a pesquisa descritiva, a qual TRIVIÑOS (2009, apud GERHARDT e SILVEIRA, 1987) o descreve como um estudo de nível intermediário, uma vez que não é tão aprofundada quanto a pesquisa explicativa nem tão alto nível quanto a exploratória.

3.2 ESTUDO DE CASO: O SISLAC

O ambiente do estudo de caso é o sistema SisLAC, mantido e administrado pela FTT da UniEVANGÉLICA. Esse sistema "visa controlar as atividades pertinentes à realização de análises clínicas do Laboratório de Análises Clínicas do Curso de Farmácia da UniEvangélica (LAC)" (FTT, 2012, p. 8). Possui diversas funcionalidades como cadastro e controle de usuários; geração de relatórios e rastreabilidades, além da disponibilização dos resultados dos exames pela *web*.

Figura 4 - SisLAC

Fonte: Fábrica de Tecnologias Turing, disponível em http://inf.unievangelica.edu.br/

3.3 ESTUDO E SELEÇÃO DAS FERRAMENTAS

3.3.1 Ferramentas de Varredura de Vulnerabilidades em Servidores

Para a obtenção de informações sobre o servidor onde o SisLAC está em execução serão utilizados os softwares *Network Mapper* (Nmap) e Nessus. O Nmap é conceituado da seguinte forma:

> Nmap significa *Network Mapper*, e é usado para fazer a varredura de *hosts* e serviços em uma rede. Nmap tem recursos avançados que podem detectar diferentes aplicações rodando em sistemas, bem como serviços e recursos de impressão digital OS[5]. É um dos *scanners* de rede mais utilizados, tornando-se muito eficaz, mas também muito detectável (MUNIZ; LAKHANI, 2013, p.59)

A vantagem principal de usar esse software é que além do Nmap ser uma poderosa ferramenta para mapeamento de rede, possui grande quantidade de técnicas automatizadas e opções disponíveis para varredura de rede (MUNIZ; LAKHANI, 2013).

Já o Nessus é uma solução para análise de vulnerabilidades mundialmente utilizada, com mais de um milhão de usuários ativos. Ele previne ataques a redes identificando vulnerabilidades e fraquezas de configurações que poderiam ser exploradas por agentes de ataque. Foi utilizado para identificar informações do servidor do SisLAC como: portas abertas, serviços em execução, versões desses serviços, versões do sistema operacional do servidor, presença de *firewall*, dentre outros (TENABLE, 2017).

A escolha das supracitadas ferramentas deu-se após uma avaliação dos recursos necessários para realização das tarefas propostas na metodologia do trabalho (Figura 5). Entre os critérios estão: facilidade de uso, popularidade da ferramenta, possibilidade de trabalho colaborativo, gratuidade, além de prover funções necessárias para coletar e mapear informações de servidores e fazer varreduras em aplicações web com alto índice de eficiência.

3.3.2 Ferramenta de Varredura de Vulnerabilidades *Web*

A ferramenta que foi utilizada para fazer a varredura das vulnerabilidades *web* será o OWASP *Zed Attack Proxy* (ZAP) v2.6.0. Essa é uma ferramenta gratuita e mantida por centenas de voluntários que permite elicitar automaticamente vulnerabilidades de segurança em

[5] Atividade de identificação do sistema operacional da máquina alvo. (MUNIZ; LAKHANI, 2013)

aplicações *web*. Alertas detalhados são gerados após as varreduras, auxiliando os testadores a aplicarem correções às vulnerabilidades encontradas. Mais informações sobre essa ferramenta podem ser encontradas no site oficial: www.owasp.org.

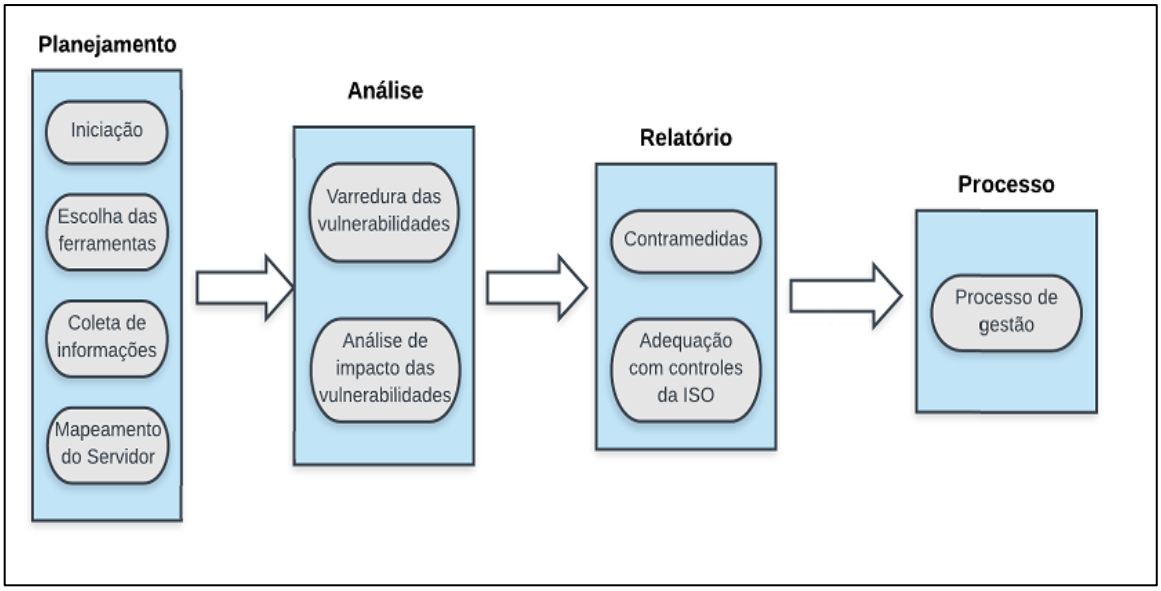
3.4 IDENTIFICAÇÃO DAS VULNERABILIDADES

Por meio de um arcabouço teórico no campo da segurança da informação, foi modelado e implementado um laboratório virtual para realização de uma análise de segurança no sistema SisLAC. Foram identificadas e classificadas as vulnerabilidades do sistema de forma a obter subsídios para, posteriormente, propor medidas para evitar ou mitigar seus riscos associados.

3.4.1 Modelo da Análise

A Figura 5 apresenta a metodologia para a análise de segurança do SisLAC. A referida análise foi realizada em um ambiente externo ao da FTT, através da execução de quatro estágios. O primeiro estágio teve como atividades a escolha das ferramentas que foram utilizadas na análise, e na coleta de informações do servidor responsável por executar a aplicação *web* SisLAC, além da delimitação do escopo, definição da janela de execução e do termo de autorização formal. O segundo foi a execução de varreduras no sistema em questão a fim de encontrar as vulnerabilidades existentes por meio das ferramentas escolhidas. O terceiro estágio foi responsável pela elaboração de um relatório com os resultados das análises e sugestões de contramedidas para extinção ou minimização dos riscos encontrados. Ao final, foi proposto um processo, por meio de um ciclo *Plan, Do, Check, Act* (PDCA), para a melhoria contínua da segurança da informação em sistemas *web*.

Figura 5 – Metodologia do trabalho



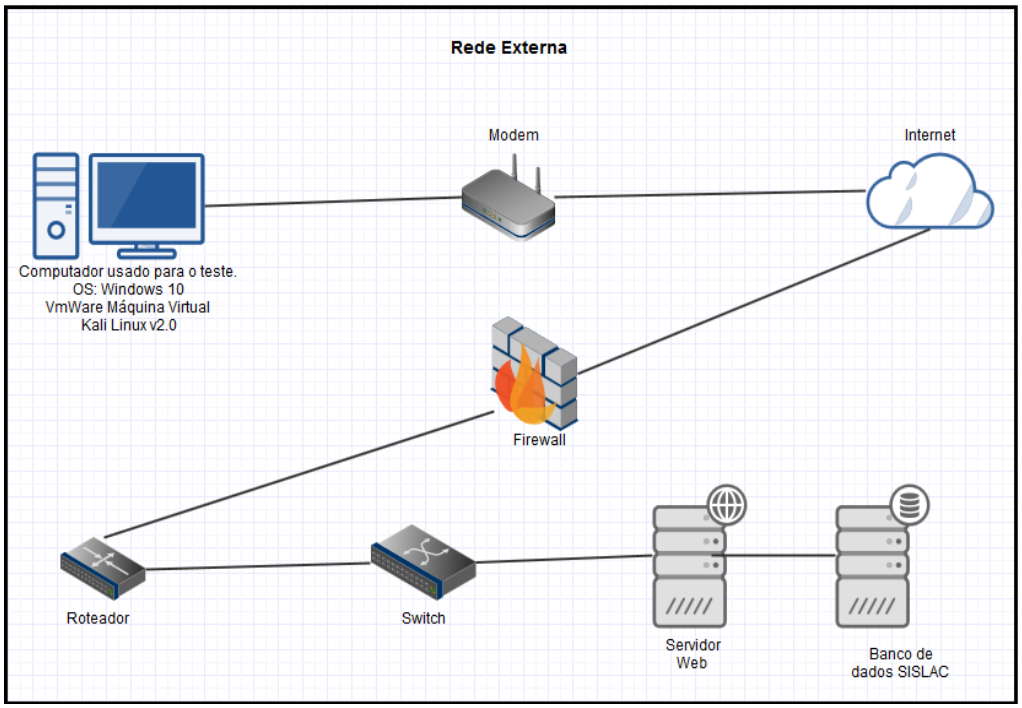
Fonte: Elaborado pelos autores.

Um processo de gestão de segurança da informação surgiu em consequência das etapas da metodologia do trabalho (Figura 5), pois ela sintetiza as etapas fundamentais, que poderão ser utilizadas periodicamente, visando a melhoria contínua da segurança da informação.

3.4.2 Detalhes da Infraestrutura

A análise supracitada foi conduzida em um ambiente virtualizado, utilizando o software VMware *Workstation* v11.0, disponível em: https://my.vmware.com. Uma máquina virtual foi criada com os seguintes recursos: 2,5 GB de memória RAM, quatro núcleos do processador Intel *CORE* i5 e 40 GB de espaço no disco rígido. O experimento seguiu uma arquitetura de rede, ilustrada na Figura 6.

Figura 6 – Arquitetura de rede



Fonte: Elaborado pelos autores.

3.4.3 Avaliação dos Riscos

O processo de avaliação de riscos teve um enfoque de alto nível, visando classificar as fraquezas encontradas na varredura de vulnerabilidades. A norma ABNT ISO/IEC 27005 destaca que:

> Uma avaliação de alto nível permite definir prioridades e uma cronologia para a execução das ações. Por várias razões, como por exemplo o orçamento, talvez não seja possível implementar todos os controles simultaneamente e, com isso, somente os riscos mais críticos podem ser tratados durante o processo de tratamento do risco (2011, p. 70).

Dentre as vantagens da aplicação dessa abordagem estão as seguintes (ISO/IEC 27005, 2011, p. 70):

- Maior probabilidade de aceitação do processo de avaliação de riscos.
- Os recursos, verbas e tempo podem ser melhor aplicados, já que a classificação prioriza os pontos mais críticos.
- Uma visão estratégica pode ser criada para auxiliar as decisões corporativas no que tange à segurança da informação.

3.4.3.1 Matriz de risco

A ordenação dos riscos a serem tratados foi obtida através da matriz (Figura 6), que apresenta uma escala de risco que foi convertida em uma classificação simples, com os seguintes índices:

- Baixo Risco: 0-2
- Médio Risco: 3-5
- Alto Risco: 6-8

Figura 7 – Matriz de riscos

	Probabilidade do cenário de incidente	Muito baixa (Muito improvável)	Baixa (Improvável)	Média (Possível)	Alta (Provável)	Muito Alta (Frequente)
Impacto ao Negócio	Muito Baixo	0	1	2	3	4
	Baixo	1	2	3	4	5
	Médio	2	3	4	5	6
	Alto	3	4	5	6	7
	Muito Alto	4	5	6	7	8

Fonte: ISO/IEC 27005 (2011, p.75).

A referida matriz mostra a relação entre a probabilidade do cenário de incidente e o impacto que pode ser causado ao negócio. O resultado dessa relação foi comparado com os três índices supracitados, sendo possível, assim, definir o nível de risco de cada vulnerabilidade.

3.5 REPORTAR O RESULTADO DA ANÁLISE

O resultado da análise foi registrado em tabela e as evidências sobre as vulnerabilidades identificadas foram dispostas conforme sua criticidade. Esses dados servem de base para a tomada de decisão quanto ao tratamento dos riscos envolvidos. O gestor diante desses dados pode selecionar as vulnerabilidades que efetivamente necessitam ser tratadas, além de escolher a melhor estratégia a ser utilizada, dentre elas: mitigar, eliminar, compartilhar ou reter os riscos (ISO/IEC 27001, 2013).

4 RESULTADOS

O principal resultado obtido foi a elaboração do processo de gestão de segurança da informação, testado em um software em execução (SisLAC).

A seguir, são descritas as análises dos resultados obtidos após a realização das atividades propostas na metodologia do trabalho.

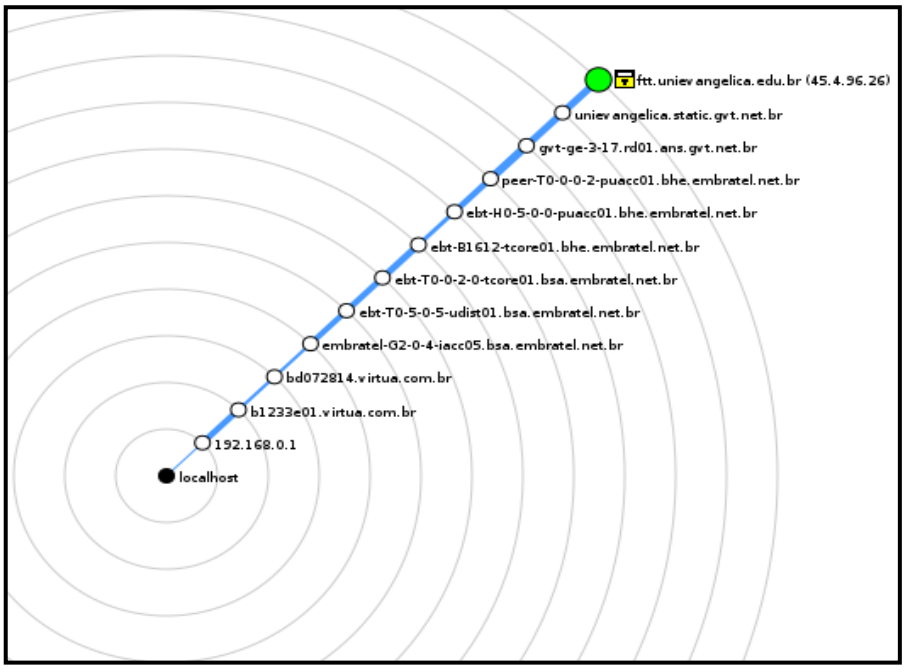
4.1 VULNERABILIDADES

4.1.1 Vulnerabilidades em Servidor Web

Para identificar possíveis vulnerabilidades em um servidor é necessário saber mais informações sobre o mesmo. Com esse pressuposto, foi feito inicialmente uma coleta de informações do servidor a fim de mapear e obter informações importantes sobre o servidor alvo.

A Figura 8 mostra a topologia de comunicação com o sistema alvo: “ftt.unievangelica.edu.br” com IP (*Internet Protocol*) “45.4.96.26”. Ela evidencia o caminho percorrido pelos pacotes até chegar ao destino (Servidor do SisLAC). A primeiro momento não houve nenhum descarte ou filtro dos pacotes enviados.

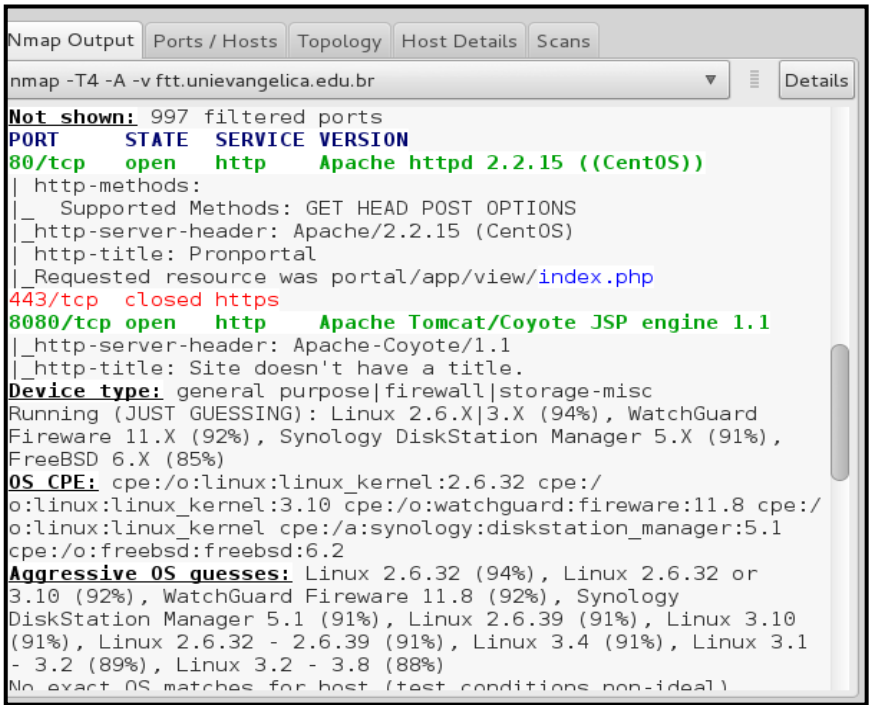
Figura 8 – Topologia de comunicação



Fonte: Captura de tela do software Zenmap no sistema operacional Kali Linux.

As Figuras 9 e 10 demonstram as saídas fornecidas pelas ferramentas Zenmap e Nessus, respectivamente.

Figura 9 – Varredura Zenmap

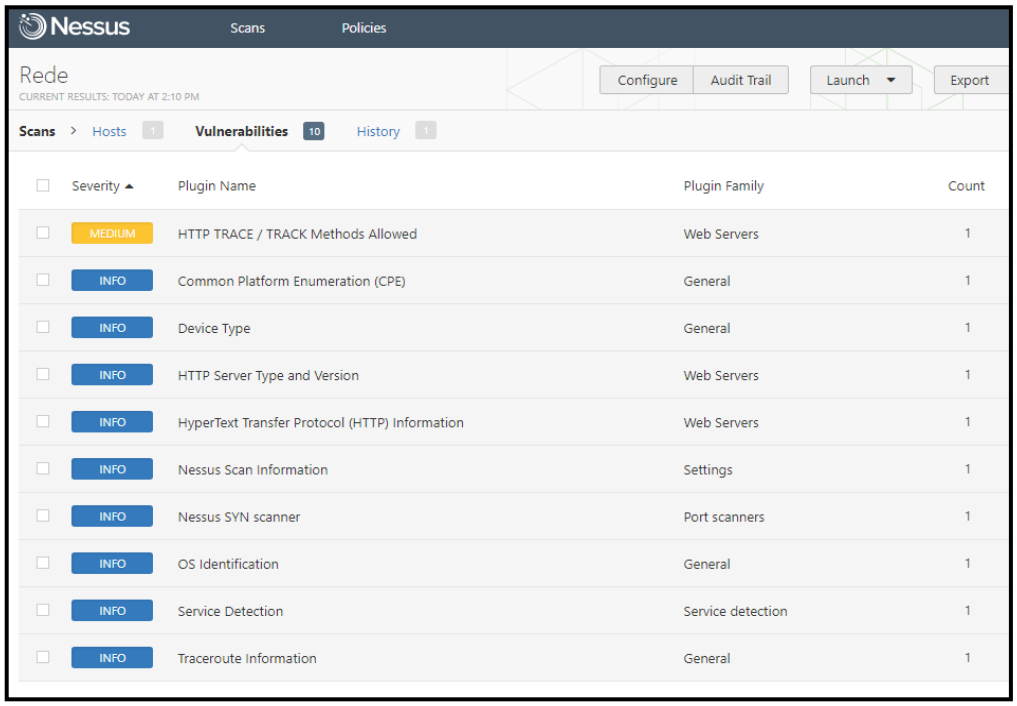


Fonte: Captura de tela de software Zenmap no sistema operacional Kali Linux.

Na interface acima, são mostrados evidências da varredura. Inicialmente são identificadas 997 portas filtradas e apenas três abertas. A porta 80/tcp encontra-se em utilização pelo serviço Apache httpd na versão 2.2.15 e rodando sobre o sistema operacional CentOS (distribuição Linux), sem a identificação da versão.

A ferramenta também destaca que a porta 443/tcp (utilizada pelo protocolo https) está fechada. Em seguida, são feitas diversas tentativas para descobrir se há e em qual versão se encontra dispositivos de segurança de rede.

Figura 10 – Varredura Nessus



Fonte:Captura de tela do software Nessus no sistema operacional Kali Linux.

Já o Nessus registra as informações encontradas de forma diferente. Informações como nome do sistema operacional e serviços em utilização foram classificados como “INFO”. A única vulnerabilidade explícita encontrada foi o fato do método HTTP TRACE estar habilitado no servidor web.

Essas ferramentas fizeram varreduras intensas (tipo de varredura completa) no sistema Alvo. Após a devida apreciação, foram identificados os seguintes alertas (Tabela 1):

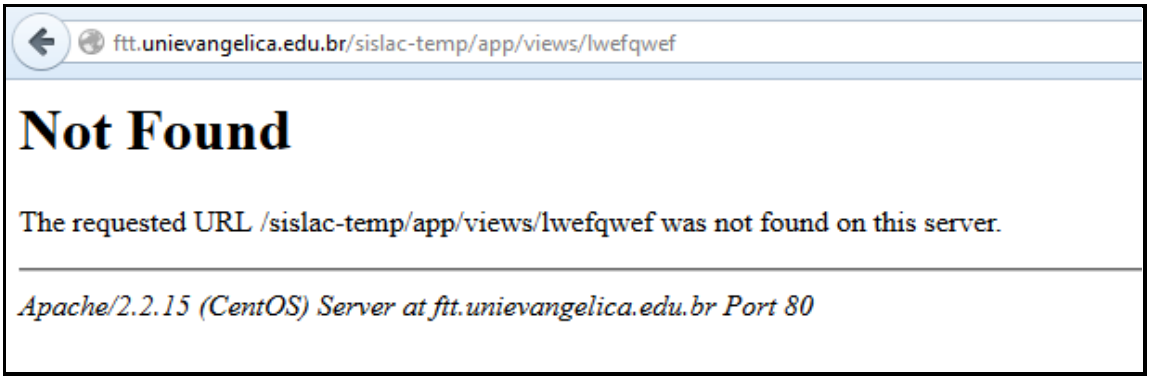
Tabela 1 – Vulnerabilidades do Servidor

Nº	Vulnerabilidade	Descrição	Nível de Risco	Impato	Probabilidade
1	Servidor Apache httpd desatualizado (Versão 2.2.15)	Foi demonstrado na Figura 2 que o lançamento de *patchs* de segurança diminuem acentuadamente o nível de risco das vulnerabilidades, porém se esses *patchs* não forem aplicados, os riscos tendem a aumentar. A versão mais recente no momento é a 2.4.28, sendo que todas as atualizações foram para correção de falhas de segurança.	6	Alto	Alto
2	HTTPS desabilitado	Utilizar HTTP, sem o protocolo SSL/TLS, faz com que os dados sejam enviados em texto plano. Isso implica na possibilidade de um Agente de ataque interceptar os dados trafegados e ler sem nenhuma dificuldade, já que não estão criptografados.	4	Médio	Médio
3	HTTP TRACE/TRACK	Esse método HTTP é habilitado por padrão e pode ser utilizado como suporte a diversos ataques, como, por exemplo, roubo de *cookies* (GROSSMAN, 2003), já que, com esse método habilitado, qualquer texto enviado ao servidor será retornado ao cliente e pode ser manipulado de forma inadequada.	2	Baixo	Muito Baixo

Fonte: Elaborado pelos autores.

Além das vulnerabilidades destacadas na Tabela 1, foi identificado uma falha de controle na aplicação do SisLAC que fornece informações indevidas sobre o servidor (Figura 11).

Figura 11 – Falha no tratamento de método HTTP



Fonte: Captura de tela do SisLAC no navegador Mozilla Firefox

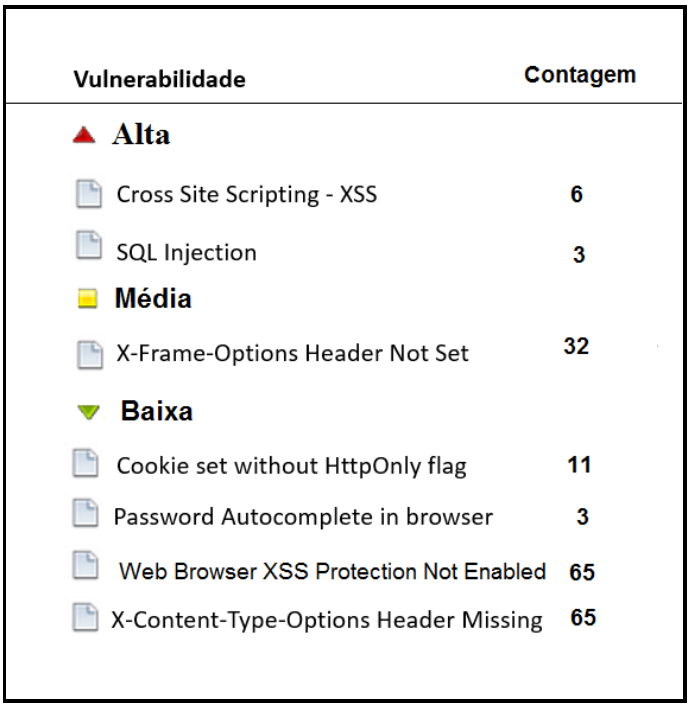
As informações contidas na imagem são obtidas quando se força como retorno um erro 404 (Não Encontrado) da aplicação. Isso ocorre por uma falha na programação segura das respostas dos métodos HTTP. Como consequência, o agente de ataque, através de uma simples adulteração na *Uniform Resource Locator* (Url*)*, pode angariar dados como nome e versão do servidor de aplicação e do nome do sistema operacional em execução no servidor alvo, dessa forma, facilitando a modelagem de ameaças.

4.1.2 Vulnerabilidades da Aplicação *Web*

Durante o período compreendido entre 28 de Junho e 30 de Junho de 2017, foram realizadas avaliações quanto à vulnerabilidades de segurança do ambiente SisLAC. A seguir, a Figura 12 revela as vulnerabilidades encontradas pela ferramenta OWASP ZAP.

As vulnerabilidades são elencadas conforme classificação pela própria ferramenta, contudo no próximo subtópico (4.1.3) essa lista de vulnerabilidades terá uma classificação diferente, dando maior destaque às vulnerabilidades de maior impacto e probabilidade de ocorrência.

Figura 12 – Vulnerabilidades Web



Fonte: Elaborado pelos autores.

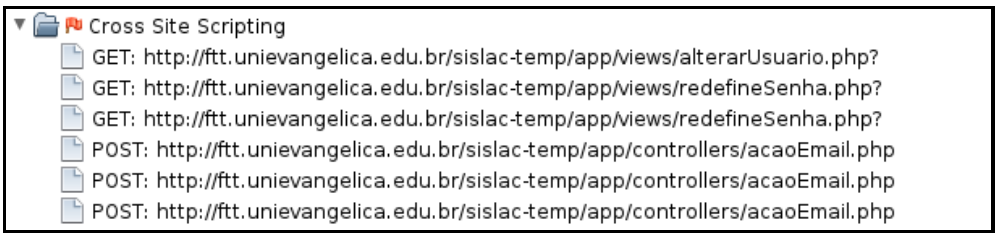
Ao total foram apontadas sete vulnerabilidades *web*, sendo que a contagem, posicionada no lado direito da imagem acima, representa a quantidade de vezes em que a vulnerabilidade foi detectada no sistema.

4.1.2.1 Vulnerabilidades Altas

As vulnerabilidades altas encontradas foram: *Cross Site Scripting* (XSS) e SQL *Injection*. Falhas como essas podem impactar diretamente na tríade da segurança da informação: confiabilidade, integridade e disponibilidade (ISO/IEC 27000, 2014).

A possibilidade de exploração da vulnerabilidade de *Cross Site Scripting* foi identificada nas seguintes páginas do sistema (Figura13):

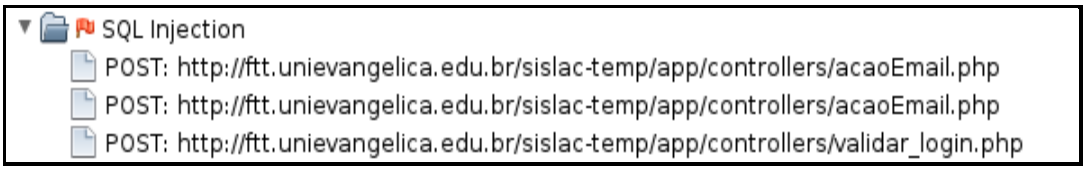
Figura 13 – Vulnerabilidades XSS



Fonte: Captura de tela de software OWASP ZAP no sistema operacional Kali Linux.

SQL *Injection* é uma das vulnerabilidades que se sustentaram durante o tempo, já que ela, mesmo sendo amplamente conhecida, continua causando graves impactos em portais e sistemas de todos os tamanhos. A exposição a essa vulnerabilidade é resultado do que a equipe de projeto e desenvolvimento produz, já que esses têm a responsabilidade de utilizar toda a expertise e técnicas para filtrar a comunicação entre os dados passados pelo usuário e o banco de dados. As raízes da vulnerabilidade de SQL *Injection* encontram-se nas seguintes páginas (Figura 14).

Figura 14 – Vulnerabilidade de SQL



Fonte: Captura de tela de software OWASP ZAP no sistema operacional Kali Linux.

4.1.2.2 Vulnerabilidade Média

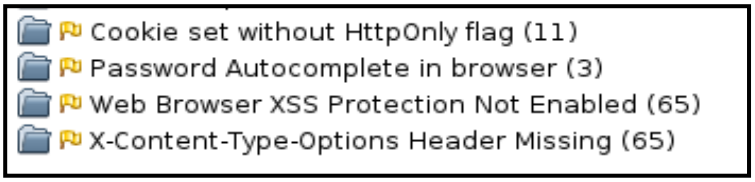
O *X-Frame-Options* não foi definido no cabeçalho das páginas do SisLAC, e isso, segundo a ferramenta, é uma vulnerabilidade de médio risco.

Navegadores atuais suportam o cabeçalho *X-Frame-Options*, que tem a função de habilitar ou desabilitar a renderização das páginas em *tags* html *frame*, *iframe* e *object*. Foi detectado que no SisLAC este cabeçalho não está configurado devidamente, dessa forma criando uma abertura para que usuários mal-intencionados usem ataques para coagir os usuários a clicarem em interfaces indesejadas (OWASP, 2017c).

4.1.2.3 Vulnerabilidades Baixas

A Figura 15 expõe as vulnerabilidades ponderadas pela ferramenta OWASP ZAP como de baixo nível de risco.

Figura 15 – Vulnerabilidades baixas



Fonte: Captura de tela de software OWASP ZAP no sistema operacional Kali Linux.

HttpOnly flag é uma opção de configuração de *cookies*[6] utilizada para informar ao navegador para não aceitar que linguagens de script acessem o "*document.cookie"*, arquivo que guarda dados de navegação do usuário. O *HttpOnly* deve ser configurado nas páginas *web* a não ser que seja necessário a manipulação desses *cookies* na aplicação (GROSSMAN, 2003).

A função de auto completar em uma entrada de dados melhora a usabilidade de um sistema *web*. Contudo, se esse atributo estiver habilitado para entradas de dados do tipo senha torna o sistema vulnerável a roubo de dados de navegação, uma vez que essas senhas precisarão ser armazenadas para que a função atinja seu objetivo (OWASP, 2017a).

A maioria dos navegadores atuais aceitam o cabeçalho *XSS-Protection* que é uma forma de filtrar ataques XSS. A sua não utilização em ambientes de produção é considerada uma fraqueza. Da mesma forma é o cabeçalho *X-Content-Type-Options* que aceita o atributo '*nosniff*', que é responsável por prevenir respostas mal interpretadas por parte dos navegadores antigos (OWASP, 2017c).

4.1.3 Ordenação e Classificação das Vulnerabilidades

A Tabela 2 consolida as vulnerabilidades analisadas, estando ordenadas de acordo com os seus níveis de riscos, obtidos através da matriz adotada (Figura 6), proveniente da ISO/IEC 27001 (2013).

Tabela 2 – Vulnerabilidades Ordenadas

OR	Vulnerabilidade	Nível	Impacto	Probabilidade
1	Injeção de SQL	**8**	**Muito Alto**	**Muito Alta**
2	*Cross-Site Scripting*	**6**	**Muito Alto**	**Média**
3	X-Frame-Options	**4**	**Médio**	**Média**
4	XSS Protection desabilitado	**3**	**Médio**	**Baixa**
5	HttpOnly flag	**2**	**Baixo**	**Baixa**

[6] Cookies são dados armazenados no computador do usuário em forma de pequenos arquivos de texto (W3SCHOOLS, 2017).

6	Password Autocomplete	**2**	**Baixo**	**Baixa**
7	X-Content-Type-Options	**1**	**Baixo**	**Muito Baixa**

Fonte: Elaborado pelos autores.

A classificação segue os parâmetros descritos na metodologia, com base nas normas internacionais, que determinam as vulnerabilidades com níveis de risco de 0 a 2 como de baixo risco, de 3 a 5 como de médio risco e de 6 a 8 como alto risco.

Importante destacar que o impacto e a probabilidade são critérios subjetivos, podendo ser diferentes dependendo do tipo negócio. O impacto relaciona-se com a criticidade das consequências potenciais e a probabilidade com a facilidade de exploração da vulnerabilidade.

Essa tabela é um instrumento importante para o tratamento dos riscos, uma vez que os gestores apenas precisam definir o nível de risco tolerável, ou seja, se há aceitação de alguma vulnerabilidade. Todas as outras devem ser tratadas. As contramedidas descritas a seguir serão úteis para esse propósito (ABNT ISO/IEC 27001, 2013).

4.1.4 Planejamento de Respostas aos Riscos

Com intuito de planejar opções de ações e cenários para responder aos riscos e à falta de controles de segurança, esse subtópico apresenta as contramedidas para as vulnerabilidades identificadas, norteando a implementação de correções, e recomendações, segundo a norma de melhores práticas em segurança da informação, para formalizar controles e diretrizes de segurança da informação na organização.

4.1.4.1 Contramedidas

Contramedida que é usada como sinônimo de salvaguarda ou controle pode ser considerada uma medida que modifica um risco, ou seja, nesse contexto, as contramedidas são ações a serem tomadas para modificar ou eliminar uma falha de segurança (ABNT ISO/IEC 27005, 2011). A Tabela 3 contém algumas contramedidas para as vulnerabilidades encontradas.

Tabela 3 - Contramedidas

Identificador	Vulnerabilidade	Contramedida
CM.01	SQL Injection	Projetistas e desenvolvedores não devem confiar nas entradas de seus usuários (OWASP, 2016). Diante disso, deve-se implementar validações para todas as entradas de dados.
CM. 02	SQL Injection	Evitar concatenação de caracteres em consultas ao banco de dados (OWASP, 2016).
CM. 03	SQL Injection	Permitir apenas o mínimo acesso possível ao banco de dados pela aplicação (OWASP, 2016).
CM. 04	Cross Site Scripting	Utilizar bibliotecas ou frameworks que previnam esse tipo de fraqueza ou tornem mais fácil a implementação de uma solução de contorno (OWASP, 2017c).
CM. 05	Cross Site Scripting	Para evitar roubo de *cookies* de usuário, deve-se assegurar que o atributo HttpOnly esteja habilitado (OWASP, 2017c).
CM. 06	Cross Site Scripting	Criar uma lista de caracteres que serão aceitos em entradas de dados de usuários, qualquer outro caractere deverá ser automaticamente rejeitado (OWASP, 2017c).
CM. 07	X-Frame-Options	Assegurar que todas as páginas estejam com esse atributo habilitado (MICROSOFT, 2010).
CM. 08	XSS Protection	Assegurar que esse filtro esteja habilitado, configurando-o para ‘1’, acionando o mecanismo de bloqueio (OWASP, 2017c).
CM. 09	HttpOnly Flag	Assegurar que o HttpOnly esteja habilitado e configurado para todos os *cookies* de sessão (OWASP, 2017a).
CM. 10	Auto completar	Assegurar que as entradas de dados que contêm senha estejam com o atributo “autocomplete” desabilitado (MICROSOFT, 2017).
CM. 11	X-Content-Type-Options	Assegurar que esse cabeçalho esteja apropriadamente configurado na aplicação. E, se possível, garantir que os usuários não utilizam a aplicação com navegadores defasados (OWASP, 2017c).

Fonte: Elaborado pelos autores.

4.1.4.2 Recomendações

Grande parte das fraquezas de segurança encontradas em sistemas *web* é resultado do trabalho de programadores e arquitetos de software que, muitas vezes, por falta de tempo hábil

ou desconhecimento deixam de tomar o devido cuidado com o desenvolvimento seguro de sistemas.

“Desenvolvimento seguro é um requisito para construir um serviço, uma arquitetura, um software e um sistema seguro” (ABNT ISO/IEC 27002, 2013, p. 81). Para atender a esse requisito, alguns aspectos devem ser considerados: a) requisitos de segurança nas fases do projeto, b) repositórios seguros, c) pontos de verificação de segurança no cronograma do projeto, d) capacidade dos desenvolvedores de conhecer, identificar e superar vulnerabilidades (ABNT ISO/IEC 27002, 2013).

A organização deve garantir que a segurança da informação faça parte de todo o ciclo de vida do desenvolvimento de sistemas de informação (ISO/IEC 27002, 2013). Dessa forma, requisitos de segurança devem ser amplamente discutidos e inseridos nos planos de desenvolvimento e melhorias de sistemas.

É necessário que haja princípios para desenvolver sistemas seguros e que eles sejam visíveis para todas as partes interessadas a fim de demonstrar sua importância.

> Convém que a segurança seja projetada em todas as camadas da arquitetura (negócios, dados, aplicações e tecnologia), realizando o balanceamento entre a necessidade da segurança da informação com a necessidade de acessibilidade. Convém que novas tecnologias sejam analisadas quanto aos riscos de segurança e o projeto seja analisado criticamente com base em modelos de ataque conhecidos. (ISO/IEC 27002, 2013, p. 85).

Por fim, convém que a segurança das aplicações em desenvolvimento e em mudança sejam avaliadas, de forma que os possíveis riscos possam ser tratados o mais rápido possível. A abrangência dessas avaliações deve ser proporcional à importância e criticidade do sistema (ABNT ISO/IEC 27002, 2013).

4.2 PROPOSTA PARA O PROCESSO DE GESTÃO DA SEGURANÇA DA INFORMAÇÃO EM SISTEMAS WEB

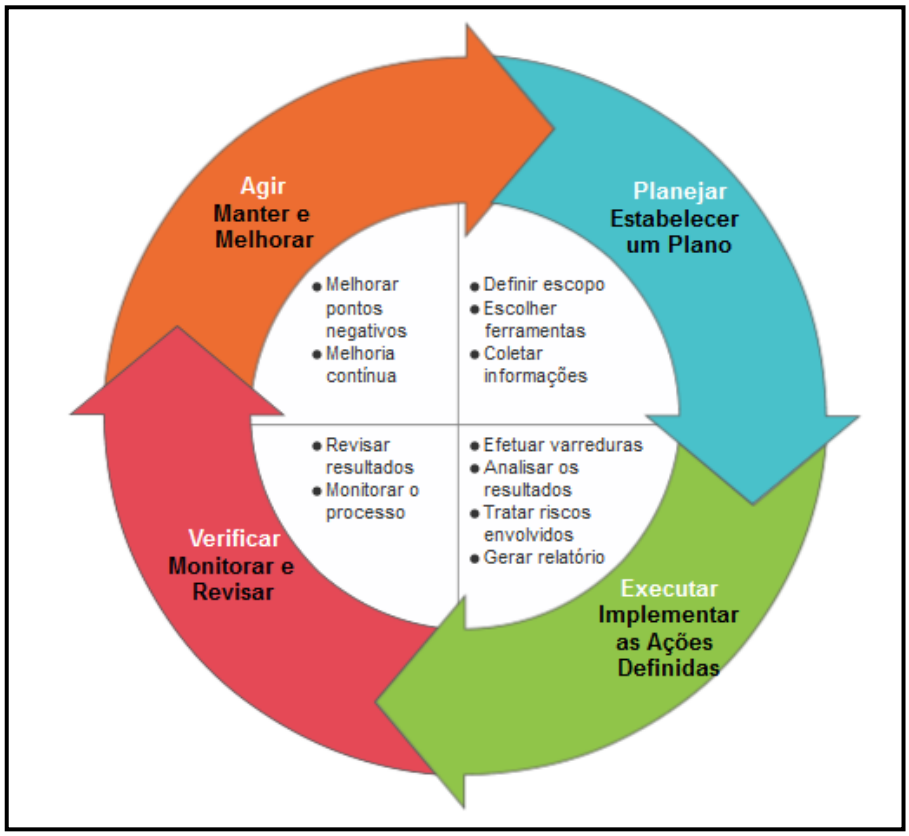
Com base no escopo do SisLAC e em normas internacionais de gestão de segurança da informação, esse subtópico ilustra e descreve um processo para a gestão da segurança da informação em sistemas *web*.

4.2.1 Visão Geral

O referido processo tem como objetivo ser instrumento contínuo de controle quanto a

segurança da informação dos sistemas *web*. Conduzindo avaliações e tratamento de riscos utilizando índices pré-definidos para tomar decisões (Figura 16).

Figura 16 – Ciclo de Vida do Processo



Fonte: Elaborado pelos autores.

4.2.2 Papéis e responsabilidades

Há três personagens relevantes e necessários para que a utilização desse processo seja eficiente. São eles: o Analista ou equipe de Analistas, a direção da organização e a equipe de desenvolvimento.

O Analista é responsável por aplicar a avaliação de segurança para identificar os riscos associados que possam impactar na confiabilidade, disponibilidade ou integridade dos dados do sistema. Deve determinar os níveis desses riscos, assim como ordená-los. Além de documentar todas as informações relevantes.

A direção da organização é responsável por coordenar e validar o ciclo de vida do processo. Deve exercer sua função de liderança assegurando que o Analista e a equipe de desenvolvimento possuam os recursos necessários para execução do processo, assegurando que

os resultados são compatíveis com o escopo aprovado e auxiliar na promoção da melhoria contínua.

Já a equipe de desenvolvimento será responsável pelo tratamento dos riscos, mas especificamente na implementação de propostas de tratamento estabelecidas pelo analista e aprovadas pela direção da organização.

4.2.3 Fluxo do processo

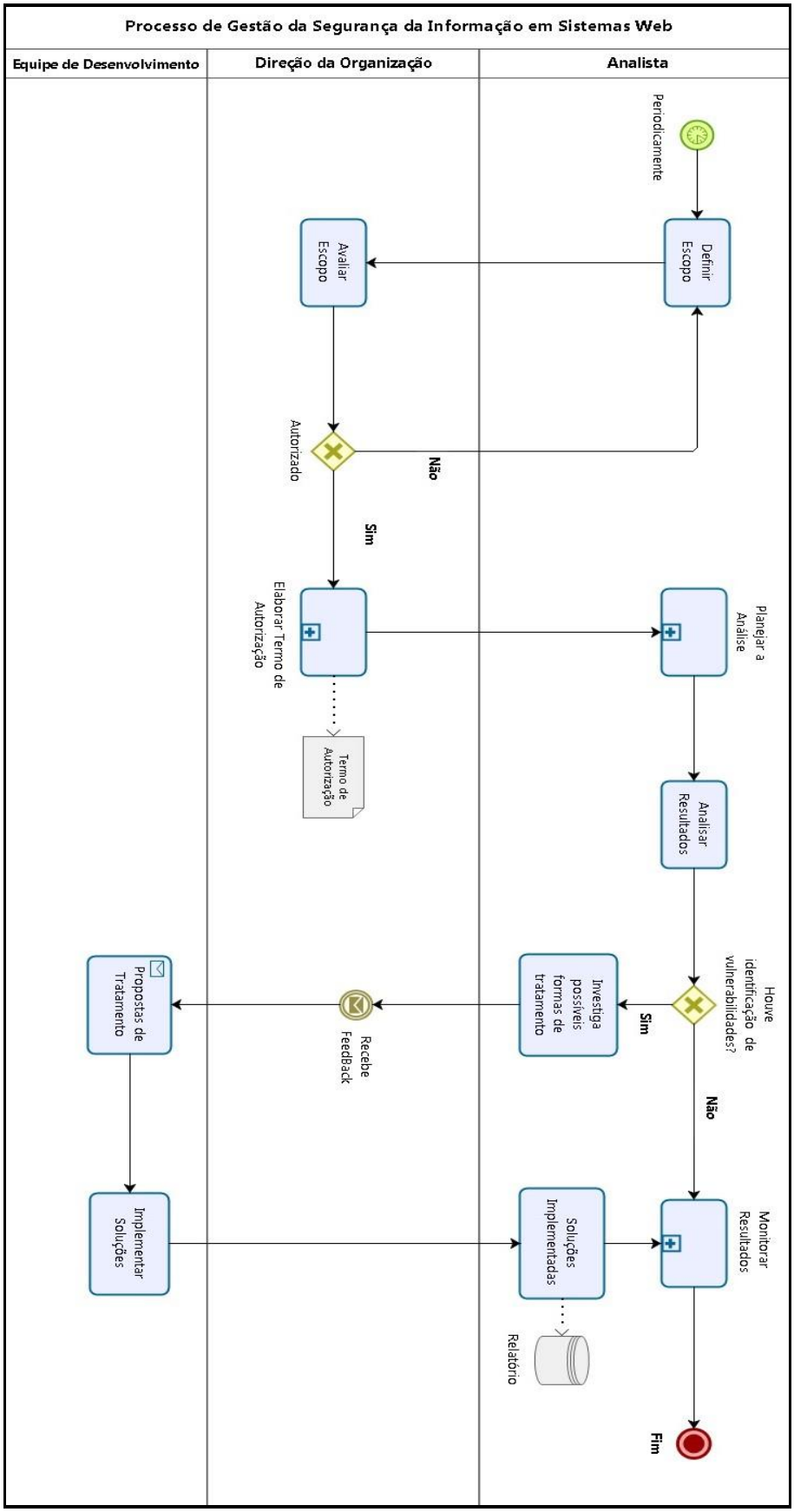
Um fluxograma foi desenvolvido para facilitar a visualização das etapas do processo de acordo com cada papel desempenhado. Foi utilizada a notação BPMN[7] para modelagem gráfica do processo, visando facilitar o entendimento e promover a padronização da visão do processo.

O processo como um todo é representado como uma piscina, e as funções do processo são representadas por raias (faixas funcionais). Já as atividades (termo genérico para trabalho executado) estão divididas em tarefas, menor nível de granularidade, e subprocessos, atividades que contém outras atividades coordenadas. São representados, ainda, artefatos de dados que representam saída de dados e a persistência desses dados em um repositório.

A Figura 17 apresenta a descrição gráfica desse fluxograma, seguida pelos seus subprocessos relacionados.

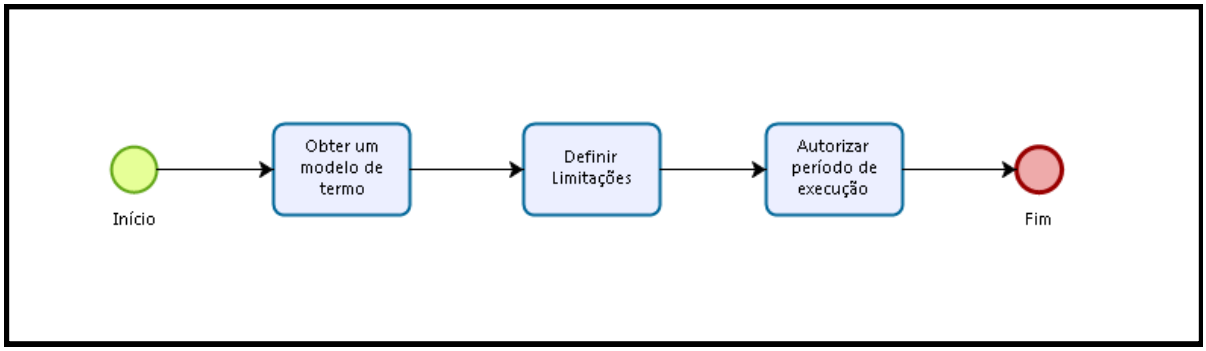
[7] Business Process Model and Notation ou Notação de Modelagem de Processos de Negócio.

Figura 17 – Fluxo do Processo



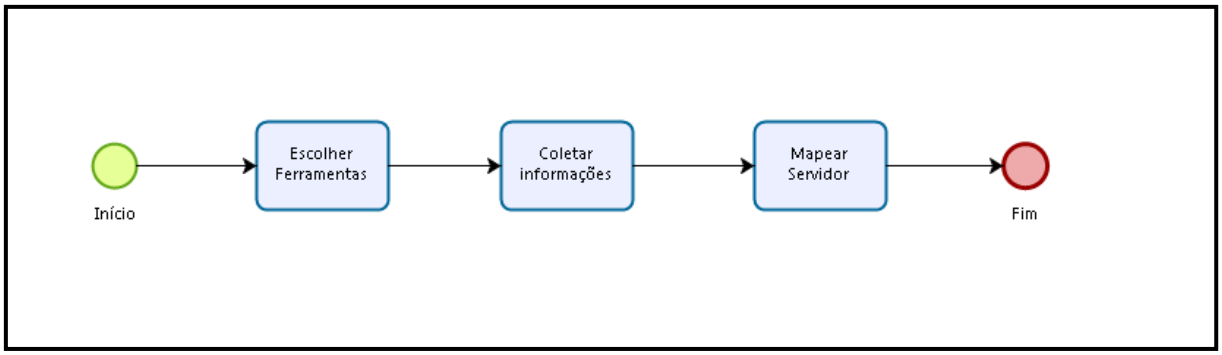
Fonte: Elaborado pelos autores.

Figura 18 – Subprocesso Elaborar Termo de Autorização



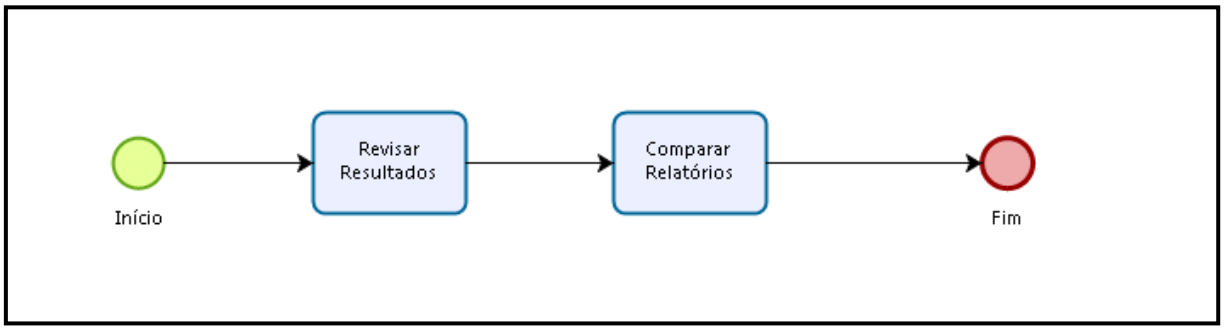
Fonte: Elaborado pelos autores.

Figura 19 – Subprocesso Planejar a Análise



Fonte: Elaborado pelos autores.

Figura 20 – Subprocesso Monitorar resultados



Fonte: Elaborado pelos autores.

4.2.4 Periodicidade do processo

A organização deve realizar essas avaliações em intervalos planejados ou quando uma mudança significativa ocorrer. A norma ainda salienta sobre a importância de se reter informação documentada dos procedimentos e resultados dessas avaliações (ABNT ISO/IEC 27001, 2013).

5 CONSIDERAÇÕES FINAIS

O desenvolvimento desse estudo possibilitou o desenho de um processo de gestão de segurança da informação em sistemas *web*, de forma a parametrizar e uniformizar a execução, monitoração e controle do trato com a segurança de tais sistemas. Além disso, foi possível, através do estudo de caso, fornecer subsídios para implementação da gestão da segurança da informação no sistema SisLAC, por meio da identificação, classificação e priorização das vulnerabilidades presentes no sistema com o auxílio de ferramentas automatizadas.

A partir de um laboratório de testes, o qual contemplou ferramentas específicas (Zenmap, Nessus e OWASP ZAP) que foram essenciais para maximização do tempo e redução de erros humanos, informações relevantes do ambiente foram levantadas a partir de varreduras para identificação de vulnerabilidades.

Uma análise das vulnerabilidades foi realizada em seguida e, como resultado, foram identificadas três vulnerabilidades no servidor *web* e sete vulnerabilidades na aplicação, sendo duas delas de alto risco. Isto demonstra que o investimento em segurança não caminha no mesmo ritmo do desenvolvimento desses produtos de software.

Os resultados da análise apontaram falhas de controles, como, por exemplo, a inobservância quanto a atualizações de seu servidor de aplicação. Isso acarreta em aberturas que podem comprometer a disponibilidade, integridade e confiabilidade do sistema. Em seguida, foram apresentadas contramedidas para eliminação ou mitigação dos riscos identificados.

Para a implantação da gestão da segurança, o primeiro passo é saber corretamente a situação atual do sistema quanto a vulnerabilidades e controles, e, em seguida, tomar medidas para solucionar eventuais problemas encontrados. Porém, sabendo da dinamicidade das vulnerabilidades na *web* e de que erros humanos podem ocorrer, foi conveniente consolidar os passos em um processo de gestão de segurança da informação visando a melhoria contínua no processo de desenvolvimento de sistemas *web*.

É importante ressaltar que o trabalho dispôs de algumas limitações. No estudo de caso, devido ao tempo limitado, não foi possível exercitar todas as fases do processo desenvolvido. Apenas o planejamento, a coleta, a análise e as propostas de contramedidas foram realizadas, fornecendo subsídios para que a implementação das próximas fases seja mais simples e intuitiva. Além do mais, na execução deste trabalho não foi analisado estaticamente o código da aplicação, que apesar de agregar valor ao experimento, aumentaria muito o escopo da

pesquisa.

Espera-se que, com os resultados obtidos em decorrência deste trabalho, os gestores da FTT possam utilizar os resultados desse estudo como um instrumento para auxiliar na aplicação de medidas de segurança durante o processo de desenvolvimento e manutenção dos produtos de software.

Em relação a trabalhos futuros, uma linha de pesquisa promissora é a criação de métricas quantitativas para a avaliação de uma organização no que tange a aderência e capacidade de execução do processo descrito neste trabalho. Outro nicho a ser explorado é a ampliação desse processo, para que contenha, de forma detalhada e direcionada, diretrizes para o desenvolvimento seguro das aplicações desenvolvidas na FTT.

REFERÊNCIAS

ACUNETIX. ***Is your website hackable*?** [s.l], 2017. Disponível em: <https://www.acunetix.com/resources/wvsbrochure.pdf>. Acesso em: 01 fev. 2017.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TECNICAS. ABNT ISO/IEC 27000. **Tecnologia da informação - Técnicas de segurança - Sistemas de gestão da segurança da informação - Visão geral e vocabulário**. Rio de Janeiro, 2014.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TECNICAS. ABNT ISO/IEC 27001. **Sistemas de gestão da segurança da informação - Requisitos**. Rio de Janeiro, 2013.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TECNICAS. ABNT ISO/IEC 27002. **Boas práticas para a gestão de segurança da informação**. Rio de Janeiro, 2013.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TECNICAS. ABNT ISO/IEC 27005. **Tecnologia da Informação – Técnicas de Segurança – Gestão de Riscos de Segurança da Informação**. Rio de Janeiro, 2011.

CENTRO DE ESTUDOS, RESPOSTA E TRATAMENTO DE INCIDENTES DE SEGURANÇA NO BRASIL. CERT.BR. São Paulo, 2012. **Cartilha de segurança para internet**. Disponível em: <https://cartilha.cert.br/livro/cartilha-seguranca-internet.pdf>. Acesso em: 4 abr. 2017.

TURING, Fábrica de Tecnologias. FTT. **Plano de Projeto** SisLAC. Anápolis, 2012.

GIAVAROTO, Sílvio; SANTOS, Gerson. **Backtrack Linux Auditoria e teste de invasão em redes de computadores**. Rio de Janeiro: Editora Ciência Moderna Ltda, 2013.

GROSSMAN, J. **Cross Site Tracing**. California, 2003. Disponível em: <http://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper_XST_ebook.pdf>. Acesso em: 17 out. 2017

GERHARDT, Tatiana; SILVEIRA, Denise. **Métodos de Pesquisa**. 1ª ed. Rio Grande do Sul: 2009.

MICROSOFT. **Disable AutoComplete for forms**. [s.l], 2017. Disponível em: <https://msdn.microsoft.com/en-us/library/ms814860.aspx>. Acesso em: 2 jun. 2017.

MICROSOFT. **Combating ClickJacking With X-Frame-Options**. [s.l], 2010. Disponível em: <https://blogs.msdn.microsoft.com/ieinternals/2010/03/30/combating-clickjacking-with-x-frame-options/>. Acesso em: 2 jul. 2017.

MUNIZ, Joseph; LAKHANI, Aamir. ***Web penetration testing with* Kali Linux**: *A practical guide to implementing penetration testing strategies on websites, web applications, and standard web protocols with* kali Linux. Birmingham: Packt, 2013.

NETCRAFT. **How many active sites are there?** Disponível em:

<http://www.netcraft.com/active-sites/>. Acesso em: 5 fev. 2017.

OWASP. **HttpOnly**. 2017. Disponível em: <https://www.owasp.org/index.php/HttpOnly>. Acesso em: 18 set. 2017a.

_____. *Open Web Application Security Project*. **OWASP Top 10 – 2017**: *The ten most critical web application security risks.* Disponível em: <https://github.com/OWASP/Top10/raw/master/2017/OWASP%20Top%2010%20%202017%20RC1-English.pdf.>. Acesso em: 20 abr. 2017b.

_____. OWASP. *Open Web Application Security Project*. **OWASP Testing Guide** v3.0. 2008.Disponível em: <https://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf>. xAcesso em: 01 fev. 2017.

_____. **XSS (Cross Site Scripting) Prevention Cheat Sheet**. 2017. Disponível em: <https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet>. Acesso em: 26 set. 2017c.

_____. **SQL Injection Prevention Cheat Sheet**. 2016. Disponível em: <https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet>. Acesso em: 4 jul. 2017.

TENABLE. **Nessus**. Disponível em: <http://www.tenable.com/products/nessus-vulnerability-scanner>. Acesso em: 14 set. 2017.

WEIDMAN, G. **Penetration testing: A hands – On Introduction to Hacking**. San Francisco: No Starch Press, 2014.

W3SCHOOLS. **Java Script Cookies**. Disponível em: <https://www.w3schools.com/js/js_cookies.asp>. Acesso em 10 out.2017.