



**LUCAS LIMA DE SOUZA**

**DESENVOLVIMENTO SEGURO DE  
APLICAÇÕES *WEB* SEGUINDO A  
METODOLOGIA OWASP**

**LAVRAS - MG  
2012**

**LUCAS LIMA DE SOUZA**

**DESENVOLVIMENTO SEGURO DE APLICAÇÕES *WEB*  
SEGUINDO A METODOLOGIA OWASP**

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Sistemas de Informação para obtenção do título de Bacharel em Sistemas de Informação.

Dr. Joaquim Quinteiro Uchôa  
Orientador

**LAVRAS - MG  
2012**

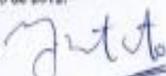
**LUCAS LIMA DE SOUZA**

**DESENVOLVIMENTO SEGURO DE APLICAÇÕES WEB  
SEGUINDO A METODOLOGIA OWASP**

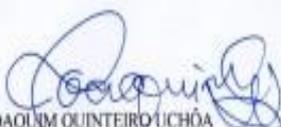
Monografia de graduação apresentada ao  
Colegiado do Curso de Sistemas de  
Informação, para obtenção do título de  
Bacharel em Sistemas de Informação.

APROVADA em 9 de outubro de 2012.

JOSÉ MONSERRAT NETO



RAPHAEL WINCKLER DE BETTI



JOAQUIM QUINTEIRO UCHÔA

(orientador/a)

LAVRAS-MG

2012

*Dedico esse trabalho aos meus pais.*

## **AGRADECIMENTO**

Deus, obrigado por ter sido meu pastor diante dessa caminhada percorrida. Sou grato pela inteligência que me deste para conhecer e por ter me ajudado a compreender melhor o mundo e as pessoas. Agradeço aos meus familiares avôs e avós, tios e tias, primos e primas, sogro, sogra e cunhado, e em especial ao meu irmão Bruno, minha mãe Hariene e meu pai Hailton, por ser meu alicerce para romper todas as barreiras e alcançar meus objetivos. Sou grato ao meu grande amor, Eduarda, pelo companheirismo, carinho, amizade e paciência durante estes anos. Enfim agradeço a todos os professores, em especial meu orientador Joaquim Quinteiro, alunos e servidores que de modo direto ou não me ajudaram a terminar mais uma *Sprint* de um grande projeto que é a vida. \o/

## RESUMO

As empresas e desenvolvedores de sistemas *Web* não apresentam como foco, utilizar orientações para prover o desenvolvimento seguro de aplicações *Web*. Nesse sentido, o presente trabalho propõe avaliar a metodologia proposta pela OWASP para o desenvolvimento seguro de aplicações *Web*. Com o objetivo de verificar como e se as recomendações dessa metodologia podem ser adotadas em projetos reais, foi realizada uma pesquisa-ação com abordagem qualitativa para conhecer de forma correta e objetiva as práticas e técnicas de segurança. Com as técnicas e métodos de segurança observados foi desenvolvida uma aplicação *Web* com as funcionalidades básicas de um *e-commerce* seguro, com objetivo de aplicar e avaliar as recomendações propostas pela OWASP. O trabalho buscou observar as relevâncias das recomendações e técnicas de segurança utilizadas e a conscientização das pessoas envolvidas no projeto.

**Palavras-chave:** aplicação *Web*, segurança, desenvolvimento seguro, *Web*, Internet e OWASP.

## SUMÁRIO

1.	INTRODUÇÃO .....	11
1.1	Contextualização e Motivação.....	11
1.2	Definição do Problema.....	12
1.3	Definição da Solução.....	12
1.4	Objetivos do Trabalho .....	13
1.5	Estrutura do Trabalho .....	13
2.	REFERENCIAL TEÓRICO.....	15
2.1	Aplicação <i>Web</i> .....	15
2.1.1	Conceito e Importância .....	15
2.1.2	<i>Web 2.0</i> .....	17
2.1.3	Segurança na <i>Web</i> .....	20
2.1.4	Vulnerabilidade e Ataques .....	22
2.2	OWASP.....	27
2.2.1	Modelagem de riscos Estruturada a Ameaças.....	28
2.3	Considerações Finais .....	34
3.	METODOLOGIA .....	35
3.1	Classificação da Pesquisa.....	35
3.2	Procedimentos metodológicos .....	36
3.3	Visão geral da aplicação.....	36
3.3.1	Descrição geral.....	36
3.3.2	Cenário .....	37
3.4	Objetivos de Segurança .....	38
3.4.1	Restrições de segurança.....	39
3.5	Tecnologias utilizadas .....	40
3.5.1	Linguagem de desenvolvimento .....	40
3.5.2	Sistema operacional.....	41
3.5.3	Banco de dados .....	41
3.5.4	Mecanismos de segurança da aplicação.....	42
3.6	Considerações finais.....	50
4.	APLICAÇÃO DESENVOLVIDA .....	51
4.1	Visão do usuário (cliente).....	51
4.2	Visão do administrador .....	53
5.	ANALISE E DISCUSSÃO .....	61
5.1	Considerações iniciais .....	61

5.2	Análise da aplicação.....	61
5.3	Considerações finais.....	64
6.	CONCLUSÃO .....	65
7.	REFERÊNCIAS BIBLIOGRÁFICAS .....	67
8.	APÊNDICE A – REQUISITOS FUNCIONAIS DO <i>E-COMMERCE</i> .....	69
8.1.1	RF1 – Cadastro de administrador.....	69
8.1.2	RF2 – Alteração de administrador.....	69
8.1.3	RF3 – Cadastro de Produto .....	69
8.1.4	RF4 – Alteração de Produto .....	70
8.1.5	RF5 – Pesquisa de Produto.....	70
8.1.6	RF6 – Compra de Produto.....	70
8.1.7	RF7 – Cadastro de usuário (Cliente) .....	71
8.1.8	RF8 – Alteração de usuário (cliente).....	71

## LISTA DE ILUSTRAÇÕES

Figura 1 - Requisição-Resposta HTTP.....	16
Figura 2 - Aplicações com conceito de <i>Web 2.0</i> . ....	19
Figura 3 - Características da informação segura.....	26
Figura 4 - Passos para modelagem de risco estruturada a ameaças. Fonte: Adaptado de (WIEMANN, 2005).....	29
Figura 5 - Gráfico de identificação de ameaças. Fonte: Adaptado de (WIESMANN, 2005). ....	33
Figura 6 - Caso de uso para um sistema de <i>e-commerce</i> básico, onde as linhas tracejadas apresentam as ações do usuário que precisam fazer <i>login</i> . ....	37
Figura 7 - Visão geral da aplicação de <i>e-commerce</i> desenvolvida pelo trabalho. ....	38
Figura 8 - Exemplo de validação dos campos na entidade Administrador .....	43
Figura 9 - Exemplo de codificação de saída com UTF-8 .....	43
Figura 10 - Exemplo de mensagem segura emitida pelo sistema.....	44
Figura 11 - Método que garante a não visualização dos dados presente nos parâmetros especificados.....	44
Figura 12 - Método que impede a passagem de comandos enviados pelos usuários. ....	45
Figura 13 - Definição do <i>token</i> de segurança, calculado a partir da sessão atual do usuário. ....	46
Figura 14 - Mapeamento indireto de objetos - Configuração das rotas, no arquivo routes.rb, para acesso dos recursos da entidade Produto.....	46
Figura 15 - Mapeamento indireto de objetos - Modo como os recursos, da entidade Produto, foram disponíveis. ....	47
Figura 16 - Configuração de gerenciamento de sessão.....	47
Figura 17 - Botão para sair do sistema, presente em todas as páginas.....	48
Figura 18 - Método de codificação de senhas. ....	48
Figura 19 - Interceptadores de acesso ao sistema e às funcionalidades destinadas ao Administrador. ....	49
Figura 20 - Filtro no Controlador na entidade Cliente para os recursos disponíveis e recursos destinados somente ao Administrador. ....	49
Figura 21 - Página inicial do sistema de comércio eletrônico. ....	51
Figura 22 - Página para adicionar produto ao carinho de compras.....	52
Figura 23 - Páginas dos produtos adicionados ao carinho de compra. ....	53

Figura 24 - Página para o Administrador acessar o sistema. ....	54
Figura 25 - Funcionalidades disponíveis ao Administrador. ....	54
Figura 26 - Listagem de todos os clientes cadastrados no sistema. ....	55
Figura 27 - Listagem de todos os Administradores do sistema. ....	56
Figura 28 - Listagem de todos os produtos cadastrados no sistema.....	57
Figura 29 - Formulário de cadastro de novos produtos com a ausência de alguns campos obrigatórios. ....	58
Figura 30 - Tentativa de remoção de algum produto no sistema. ....	59
Figura 31 - Página para edição de Produto.....	60

## 1. INTRODUÇÃO

### 1.1 Contextualização e Motivação

Atualmente a Internet contém milhões de computadores conectados em uma rede física, formando um dos sistemas mais complexos da atualidade. Um computador ligado à Internet possui centenas de programas de *software* em execução que interagem uns com os outros (SCHNEIER, 2011). Esse relacionamento permite uma disseminação e divulgação de informação no âmbito mundial, proporcionando a colaboração e interação entre indivíduos e os seus computadores, independente de sua localização geográfica.

A evolução da Internet proporcionou uma mudança importante que passou a ser percebida na construção de aplicações *Web*. Neste sentido, para Lawton (2007), páginas que apresentavam um comportamento estático passaram a ser interativas e dinâmicas. Essa mudança proporcionou aos usuários a capacidade de publicar e consumir informação de forma rápida e constante, em aplicações como redes sociais<sup>1</sup>, *wikis*<sup>2</sup>, *blogs*<sup>3</sup> e outros.

Nesse contexto, o desenvolvimento seguro de uma aplicação *Web* tornou-se um componente crucial para os sistemas de *software* no mercado. Isso por que a popularização de aplicações *Web* com vulnerabilidade (que permite um invasor violar a integridade do sistema) também aumentou, de modo a intensificar os problemas de segurança na *Web*. Neste sentido, os arquitetos de aplicações *Web* não devem se responsabilizar apenas por modelar uma aplicação, mas também pela construção de projetos que resistem aos riscos, lidando com o uso típico das aplicações, porém preocupando-se com os riscos de segurança mais críticos.

---

<sup>1</sup> Estrutura social na *Web* com o objetivo de compartilhar informações, gostos e ideias entre usuários com o mesmo estilo.

<sup>2</sup> *Software* colaborativo que permite a edição coletiva de documentos em hipertexto.

<sup>3</sup> Diário eletrônico que permite atualização de conteúdos de maneira dinâmica.

De acordo com Wiesmann *et al.* (2005), não existe ainda na maioria das empresas e projetistas de *software*, uma movimentação forte em prol da segurança de TI (Tecnologia da Informação), ou seja, as empresas não têm adotado, em nível adequado para auxiliar o desenvolvimento seguro de aplicações, atividades como controle de ameaças, revisão de código e análise de riscos de segurança. O mesmo autor sugere que as recomendações propostas pelas metodologias de desenvolvimento de segurança de aplicações *Web*, suportem os objetivos empresariais e auxiliem os projetistas de *software* e desenvolvedores a construir uma aplicação segura, de modo que não se utilize apenas os conceitos básicos, mas também que haja a valorização dos princípios de segurança como: confidencialidade, integridade, autenticidade e disponibilidade.

## **1.2 Definição do Problema**

Diante da falta de orientações para realizar o desenvolvimento seguro de aplicações *Web*, define-se o problema deste trabalho como: *A maioria dos arquitetos e desenvolvedores não utilizam, durante o ciclo de vida de projetos de software, orientações e técnicas para prover o desenvolvimento seguro de aplicações Web, identificando os objetivos de segurança no sistema de software, e seguindo os princípios como: confidencialidade, autenticidade, integridade e disponibilidade dos dados.*

## **1.3 Definição da Solução**

Para buscar informações que possam contribuir com o problema identificado, deve-se: *Realizar estudos para qualificar a adoção de técnicas e recomendações de segurança no desenvolvimento de aplicações Web através de metodologias para o tratamento de riscos e ameaças de segurança contida em um projeto real.*

#### 1.4 Objetivos do Trabalho

Neste trabalho, definiu-se como objetivo geral: *Avaliar a metodologia proposta pela OWASP para o desenvolvimento seguro de aplicações. Verificar como e se as recomendações dessa metodologia podem ser adotadas em projetos reais.*

Os seguintes objetivos específicos foram definidos, a fim de alcançar o objetivo geral proposto:

1. Pesquisar livros e artigos, com conceitos relacionados a este trabalho;
2. Pesquisar as maneiras de ataques e vulnerabilidades mais comuns no ambiente *Web*;
3. Estudar o Guia para Desenvolvimento Seguro de Aplicações *Web* e *Web Service*, WIESMANN *et al.* (2005);
4. Analisar o *The Ten Most Critical Web Application Security Risks* (OWASP, 2010);
5. Avaliar qualitativamente as recomendações e técnicas de segurança presente no Guia para Desenvolvimento Seguro de Aplicações *Web* e *Web Service*, por meio de simulações de ataque em um sistema básico de compras *online*.

#### 1.5 Estrutura do Trabalho

Capítulo 2 apresenta o referencial teórico, mostrando os principais conceitos utilizados no trabalho como aplicação *Web*, *Web 2.0*, segurança na *Web*, vulnerabilidades e ataques, pilares da informação segura. É apresentada também a metodologia OWASP, com conceitos de modelagem de risco estruturada a ameaças, e falhas em aplicações *Web*.

Capítulo 3 mostra a metodologia do trabalho, detalhando a sequência de atividades, a maneira de condução do trabalho e o modo como os conceitos serão aplicados para alcançar o resultado esperado.

Capítulo 4 detalha as funcionalidades e o comportamento do protótipo de *software* desenvolvido, apresentando as principais funcionalidades e como utiliza-las.

Capítulo 5 apresenta uma análise e discussão sobre a metodologia e técnicas utilizadas para a construção do protótipo de um sistema de compra, considerando os conceitos e recomendações de segurança.

Capítulo 6 apresenta a conclusão das atividades e conceitos relatados no trabalho.

## 2. REFERENCIAL TEÓRICO

### 2.1 Aplicação *Web*

#### 2.1.1 Conceito e Importância

No início da década 1990, *Tim Berners-Lee* apresentou ao mundo a *World Wide Web* ou *Web*. *Web* é um sistema computacional projetado para ser acessado por meio de uma rede como a Internet ou Intranet. Tecnicamente trata-se da utilização do protocolo HTTP – Protocolo de Transferência de Hipertexto, que é implementado em dois programas: um programa cliente e outro servidor, para conversar um com o outro por meio da troca de mensagens HTTP (WANG, 2011).

Um recurso *Web* (como páginas, imagens e vídeos) é acessado por um programa cliente. O programa cliente, normalmente conhecido como navegador ou *browser* (por exemplo, Mozilla Firefox<sup>4</sup>, Opera<sup>5</sup> e Chrome<sup>6</sup>), que acessam os recursos através de um esquema de endereçamento uniforme conhecido como URL – *Uniform Resource Locator*. Os servidores *Web*<sup>7</sup> abrigam os recursos *Web*, cada um endereçado por uma URL.

De acordo com Kurose (2010), a maneira como os clientes *Web* requisitam páginas aos servidores e como os servidores as transferem a clientes é estabelecida pelo HTTP. Quando um usuário requisita uma página *Web* (por exemplo, digita uma URL no navegador), o *browser* envia ao servidor mensagens de requisição HTTP para os recursos das páginas. O servidor recebe as requisições e responde com mensagens de respostas HTTP com os recursos solicitados. A Figura 1 ilustra o comportamento de requisição-resposta do HTTP.

---

<sup>4</sup> Navegador Mozilla Firefox: [http:// www.mozilla.org/pt-BR/firefox/](http://www.mozilla.org/pt-BR/firefox/)

<sup>5</sup> Navegador Opera: <http://www.opera.com/>

<sup>6</sup> Navegador Chrome: <http://www.google.com/chrome/>

<sup>7</sup> Exemplo de servidor *Web* é o Apache Tomcat: <http://www.tomcat.apache.org/>

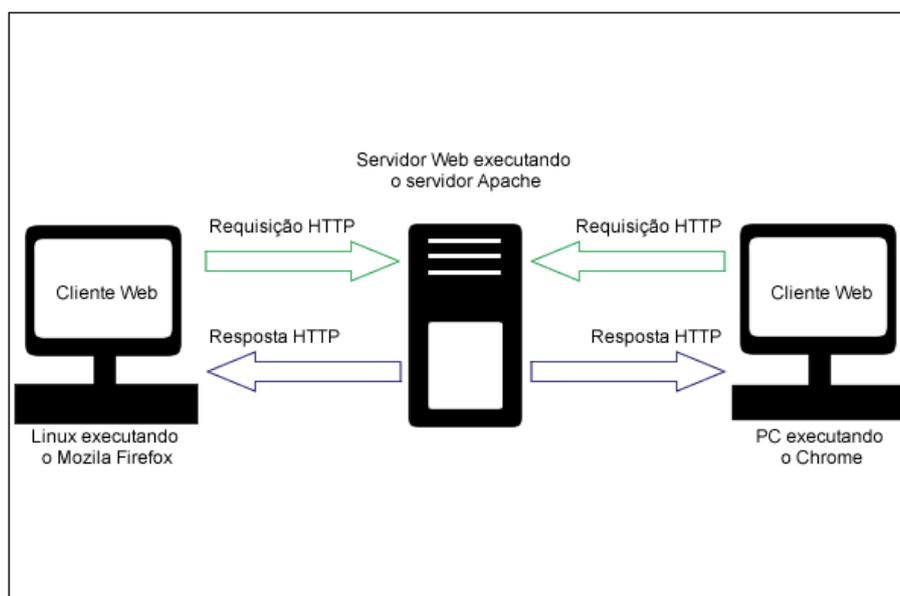


Figura 1 - Requisição-Resposta HTTP.

Geralmente a estrutura de uma aplicação de *software* que utiliza a *Web*, por meio de um *browser* como ambiente de execução e realiza o processamento em algum servidor, é dividida em três camadas. A primeira, no qual está presente no programa cliente, é denominada como camada de apresentação. A segunda, de aplicação (intermediária) possui toda a estrutura lógica; e a terceira, a camada de armazenamento, que contém as aplicações de banco de dados.

Para o desenvolvimento das duas primeiras camadas pode utilizar-se de tecnologias como PHP<sup>8</sup>, HTML<sup>9</sup>, JSP<sup>10</sup>, Flash<sup>11</sup>, Python<sup>12</sup>, Ruby on

---

<sup>8</sup> Linguagem PHP: <http://www.php.net>

<sup>9</sup> Linguagem HTML: <http://www.w3.org/MarkUp/>

<sup>10</sup> Tecnologia JSP: <http://www.oracle.com/technetwork/java/javae/jsp/index.html>

<sup>11</sup> Plataforma Adobe Flash: <http://www.adobe.com/br/products/flashplayer.html>

<sup>12</sup> Linguagem Python: <http://www.python.org>

Rails<sup>13</sup>, ASP<sup>14</sup>, CSS<sup>15</sup>, ASP.NET<sup>16</sup>, Java<sup>17</sup> etc. Na camada de armazenamento os aplicativos de banco de dados utilizados são MySQL<sup>18</sup>, Oracle<sup>19</sup>, Firebird<sup>20</sup>, Microsoft SQL Server<sup>21</sup>, PostgreSQL<sup>22</sup> entre outros. A alta gama de tecnologias proporciona aos desenvolvedores, uma alta flexibilidade para adequar a melhor tecnologia aos principais objetivos da aplicação. As aplicações construídas podem atender varias necessidades sociais e comerciais, e proporcionar aos usuários um ambiente interativo, rápido e colaborativo.

Atualmente a *Web* é uma plataforma com grande potencial para execução de tecnologias com vários serviços e informações compartilhadas que podem ser acessados de qualquer lugar do mundo, muitas vezes sem a necessidade de instalação de programas ou armazenamento de dados.

### 2.1.2 *Web 2.0*

A evolução da Internet e a mudança na forma como a *Web* passou a ser vista por usuários e desenvolvedores, em 2004, foi nomeada como a segunda geração de comunidades e serviços na *Web*, ou seja, *Web 2.0* (MURUGESAN, 2007). Embora o termo expresse uma nova versão para a *Web*, este não se refere à atualização nas especificações técnicas, mas sim o ambiente de interação e participação.

---

<sup>13</sup> Framework *Web* Ruby on Rails: <http://rubyonrails.org>

<sup>14</sup> Linguagem ASP: <http://www.asp.net>

<sup>15</sup> Folha de estilo CSS: <http://www.w3.org./Style/CSS>

<sup>16</sup> Linguagem ASP.net : <http://www.asp.net>

<sup>17</sup> Linguagem Java: <http://home.java.net>

<sup>18</sup> Banco de Dados MySQL: <http://www.mysql.com>

<sup>19</sup> Banco de Dados Oracle: <http://www.oracle.com/br/products/database/index.html>

<sup>20</sup> Banco de Dados Firebird: <http://www.firebirdsql.org/>

<sup>21</sup> Banco de Dados Microsoft SQL Server: <http://www.microsoft.com/sqlserver>

<sup>22</sup> Banco de Dados PostgreSQL: <http://www.postgresql.org/>

Para Lawton (2007), páginas que apresentavam um comportamento estático<sup>23</sup> passaram a ser interativas. O mesmo autor afirma que essa interação proporcionou aos usuários, a capacidade de publicar e consumir conteúdo na Internet de forma rápida e constante, em aplicações como *blogs*, *wikis*, redes sociais e outros.

A *Web 2.0* propõe que as aplicações *Web* apresentem comportamento semelhante aos aplicativos *desktop*<sup>24</sup>, para aumentar a velocidade e a facilidade de uso das informações compartilhadas. Este propósito da *Web 2.0* poder ser claramente observado em aplicativos como GoogleDocs<sup>25</sup>, YouTube<sup>26</sup> e GoogleCalendar<sup>27</sup> como apresentado pela Figura 2.

---

<sup>23</sup> Geralmente são páginas construídas somente com HTML, sem efeitos ou funcionalidades avançadas.

<sup>24</sup> São *softwares* que funcionam em computadores pessoais sem a necessidade de um navegador de internet.

<sup>25</sup> GoogleDocs: <http://docs.google.com/>

<sup>26</sup> YouTube: <http://www.youtube.com/>

<sup>27</sup> GoogleCalendar: <http://www.google.com/calendar>

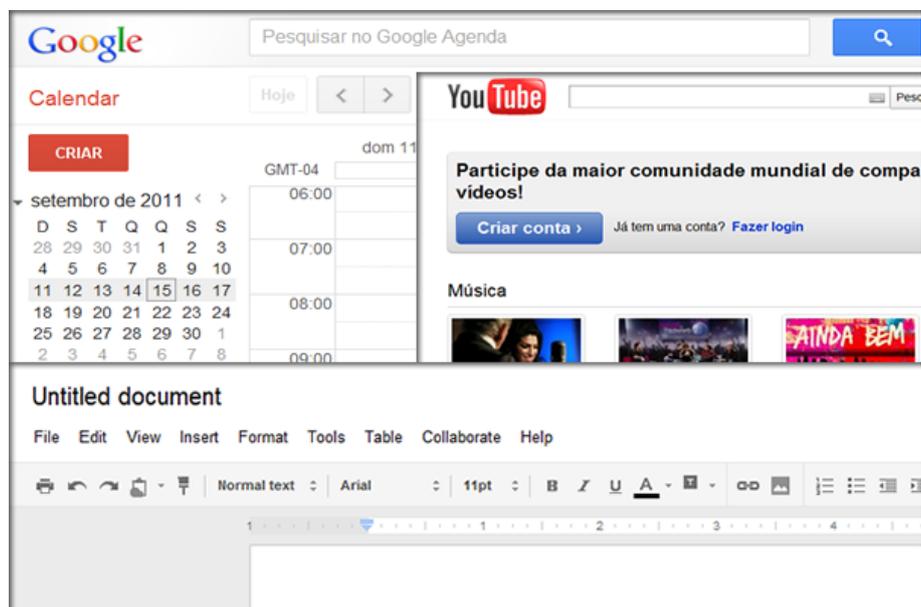


Figura 2 - Aplicações com conceito de *Web 2.0*.

Embora *Web 2.0* tenha começado simplesmente como um fenômeno de consumo, atraindo vários usuários e colaboradores para aplicações sociais, como MySpace<sup>28</sup>, Flickr<sup>29</sup>, YouTube e a enciclopédia *online* Wikipedia<sup>30</sup>, ela tem impactado de maneira relevante outras áreas de aplicação. De acordo com MURUGESAN (2007), a *Web 2.0* pode ajudar as empresas no desenvolvimento de produtos, pesquisas de mercado, aquisição de informações relevantes, publicidade e inteligência competitiva, gerando um número crescente de empresas que oferecem serviços *online* de maneira inovadora e colaborativa.

---

<sup>28</sup> MySpace: <http://www.myspace.com/>

<sup>29</sup> Flickr: <http://www.flickr.com/>

<sup>30</sup> Wikipedia: <http://www.wikipedia.org/>

### 2.1.3 Segurança na Web

Junto com a ascensão da Internet como meio de troca de dados, conteúdos e de comunicação, apareceram os recursos maliciosos<sup>31</sup> (KUROSE, 2010). Aplicações com comportamento dinâmico e interativo exigem a capacidade de interpretação de *scripts*<sup>32</sup>, os quais podem executar códigos ou transportar conteúdo inadequado, para propagar *malwares*<sup>33</sup> e *links*<sup>34</sup> mal intencionados. Esse tipo de aplicação Web apresenta maiores riscos que os tradicionais *sites Web*, pois a propagação dos recursos maliciosos ocorre através das páginas Web, mensagens de *e-mail*, arquivo MP3<sup>35</sup>, chamadas telefônicas e vídeos em tempo real (LAWTON, 2007).

Um estudo que mostra o crescimento da utilização da Web pelos *crackers*<sup>36</sup> para lançar *worms*<sup>37</sup> que executam operações prejudiciais foi feito em uma análise do Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil - CERT.br<sup>38</sup> (NIC.BR, 2011). De acordo com essa análise, as notificações sobre ataques a servidores Web, no primeiro trimestre de 2011, cresceram 14% em relação ao trimestre anterior e 69% em relação ao mesmo período de 2010. A análise ainda demonstra que os atacantes exploram vulnerabilidades em aplicações Web para, então, hospedar nesses sites páginas falsas de instituições financeiras, cavalos de tróia<sup>39</sup>, injeção de código SQL<sup>40</sup>, ferramentas utilizadas em ataques a outros

---

<sup>31</sup> Links, imagens, vídeos ou páginas que podem prejudicar os usuários.

<sup>32</sup> Um programa ou uma sequência de instruções interpretadas por outro programa.

<sup>33</sup> *Software* destinado a infiltrar em um sistema computador alheio de forma ilícita.

<sup>34</sup> É uma referência a um documento em hipertexto a outras partes deste documento ou a outro documento.

<sup>35</sup> MPEG Audio Layer-3, tipo de compressão de áudio.

<sup>36</sup> São indivíduos que invadem sistemas de computadores, a fim de fraudar os códigos de segurança.

<sup>37</sup> Um worm (verme, em português), em computação, é um programa auto replicante, não precisa de outro programa para se propagar.

<sup>38</sup> Cert.br: <http://www.cert.br/>

<sup>39</sup> Programa malicioso que facilita uma possível invasão em um computador.

<sup>40</sup> Esse assunto será tratado na sessão 2.1.4.

servidores *Web*, execução maliciosa de arquivos, e *scripts* para envio de *spam*<sup>41</sup> ou *scam*<sup>42</sup>.

Se o resultado da pesquisa realizada pela CERT.br são relevantes, é importante destacar que essa análise é apenas a parte visível do problema, uma vez que a maior parte dos ataques não são divulgados ou documentados. As instituições ocultam ou relatam poucas informações sobre ataques efetuados, para evitar o dano à imagem, e principalmente para evitar a saída de clientes que se sentiriam receosos de informar ou enviar informações sensíveis para empresas que tiveram sistemas computacionais invadidos (UCHÔA, 2011).

Se por um lado há um alto volume de usuários virtuais, mal intencionados, utilizando a Internet de maneira inadequada, por outro lado, existem agentes de segurança trabalhando a favor da segurança na *Web*, assunto destacado, na reportagem no *site* Info<sup>43</sup>, “Brasil terá Centro de Defesa Cibernética<sup>44</sup>”.

Segundo CAMPI (2011), há mais de um ano o Brasil está desenvolvendo o Centro de Defesa Cibernética (CDCiber). O objetivo do CDCiber será coordenar as ações de defesa cibernética e proteger as redes militares e governamentais, além de possibilitar uma contribuição na proteção às infraestruturas de informação.

De um modo geral a falta de segurança em sistemas *Web* ainda prevalece, porém é importante destacar que a cada dia melhorias computacionais voltadas para a segurança estão sendo implantadas. Estas

---

<sup>41</sup> Mensagens eletrônicas não solicitadas enviadas em massa.

<sup>42</sup> Mensagens eletrônicas oferecendo promoções ou vantagens, solicitando algum tipo de cadastramento.

<sup>43</sup> Site Info Abril: <http://info.abril.com.br/>

<sup>44</sup> Notícia disponível em: <http://info.abril.com.br/noticias/ti/brasil-tera-centro-de-defesa-cibernetica-09062011-21.shl>

melhorias podem ser observadas em sistemas de *e-commerce*<sup>45</sup>, *e-learning*<sup>46</sup>, *e-business*<sup>47</sup>, transações eletrônicas, redes sociais e sistemas governamentais. As melhorias apresentam como objetivo suprir a necessidade de aplicações seguras no mercado da Internet e permitir aos usuários que ainda possuem receio de utilizar serviços *Web* sintam-se seguros.

#### 2.1.4 Vulnerabilidade e Ataques

As vulnerabilidades e ataques em aplicações *Web* estão diretamente relacionados. Geralmente para um sistema computacional ser atacado é necessário que o usuário mal intencionado procure ou perceba uma fraqueza no sistema. Falhas em sistemas de Internet permite a um invasor violar a integridade do mesmo, principalmente por: uso de senhas fracas e de *bugs*<sup>48</sup> em *software* e/ou falhas no desenvolvimento da aplicação.

De acordo com a pesquisa realizada por (OWASP, 2010), no ano de 2010, os principais tipos de ataques e vulnerabilidades presentes em aplicações *Web* foram:

- **Falha de Injeção:** estas acontecem quando os dados que o usuário fornece de entrada são enviados como parte de um comando ou consulta. Os atacantes confundem o interpretador para o mesmo executar comandos manipulados, enviando dados modificados. As falhas de Injeção habilitam o atacante criar, ler, atualizar ou apagar arbitrariamente qualquer dado disponível para a aplicação. No pior

---

<sup>45</sup> É a venda não presencial de produtos feita através de um dispositivo eletrônico, como, por exemplo, um computador.

<sup>46</sup> É equivalente a um modelo de ensino, auxiliado por uma tecnologia, onde os envolvidos não estão presentes.

<sup>47</sup> São negócios efetuados entre dois ou mais comerciantes através da Internet.

<sup>48</sup> Erro no funcionamento comum de um *software*.

cenário, estas falhas permitem ao atacante comprometer completamente a aplicação e os sistemas relacionados.

- **Cross-Site Scripting (XSS):** XSS é um subconjunto de inserções HTML<sup>49</sup>. Esse tipo de ataque permite que o agente atacante execute *script* no navegador da vítima, com intenção de sequestrar sessões de usuários, desfigurar *Web sites*, inserir conteúdo hostil, conduzir ataques de roubo de informações pessoais (*phishing*) e obter o controle do navegador do usuário usando um *script* mal intencionado (*malware*). O *script* malicioso é frequentemente escrito em JavaScript, mas qualquer linguagem de *script* suportada pelo navegador da vítima é um alvo potencial para este ataque.
- **Falha de Autenticação e Gerenciamento de Sessão:** Autenticação e gerência de sessão apropriada são críticas para a segurança na *Web*. Falhas nesta área geralmente envolvem falha na proteção de credenciais e nos *tokens*<sup>50</sup> da sessão durante seu tempo de vida. Estas falhas podem estar ligadas a roubo de contas de usuários ou administradores, causando violações de privacidades.
- **Referência Direta Insegura a Objetos:** Uma referência direta a um objeto acontece quando um desenvolvedor expõe uma referência a um objeto de implementação interna, como por exemplo, um arquivo, diretório, registro na base de dados ou chave, uma URL ou um parâmetro de um formulário. Um atacante pode manipular diretamente referências a objetos para acessar outros objetos sem autorização, a não ser que exista um mecanismo de controle de acesso.
- **Cross Site Request Forgery (CSRF):** tem como objetivo iniciar, através do navegador de uma possível vítima, uma requisição para uma aplicação *Web* vulnerável. A utilização de um usuário (vítima)

---

<sup>49</sup> Linguagem de Marcação de Hipertexto para construção de páginas *Web*.

<sup>50</sup> Um conjunto de caracteres com um significado coletivo.

intermediário acontece, pois o usuário mal intencionado realiza um ataque em nome da vítima. Este ataque é também conhecido por outros diversos nomes incluindo *Session Riding*, Ataques *One-Click*, *Cross Site Reference Forgery*, *Hostile Linkig* e *Automation Attack*.

- **Gerenciamento Inseguro de Configuração:** Configurações inseguras e mal definidas em *frameworks*<sup>51</sup>, servidor de aplicações, *Web service*<sup>52</sup> e servidor de banco de dados podem resultar na violação da segurança das aplicações desenvolvidas e o fornecimento de informações inseguras.
- **Armazenamento Criptográfico Inseguro:** Um atacante pode aproveitar essa vulnerabilidade e tentar obter acesso aos conteúdos de dados sensíveis à aplicação. Simplesmente não criptografar<sup>53</sup> dados sensíveis é muito comum. Ainda, aplicações que adotam criptografia frequentemente possuem algoritmos mal concebidos, usam mecanismos de codificação inapropriados ou cometem sérios erros usando codificações fortes.
- **Falha de Restrição de Acesso à URL:** Com essa vulnerabilidade é possível que um atacante tenha acesso a páginas ou informações que não deveriam ser permitidas, caso o controle de restrição de acesso não seja desenvolvido de forma correta.
- **Proteção Insuficiente da Camada de Transporte:** Frequentemente as aplicações *Web* falham para autenticar, criptografar, e proteger a confidencialidade e integridade na rede. Quando o fazem,

---

<sup>51</sup> Conjunto de códigos comuns entre vários projetos de *software* provendo uma funcionalidade genérica.

<sup>52</sup> Solução utilizada na integração de sistemas e na comunicação de aplicações diferentes.

<sup>53</sup> Técnica pela qual os dados podem ser transformados da sua forma original para outra ilegível, de forma que possa ser conhecida apenas por seu destinatário.

geralmente utilizam de certificados<sup>54</sup> inválidos, algoritmos fracos ou não os usam corretamente.

- **Redirecionamento Inválido:** Aplicações *Web* frequentemente redirecionam e enviam os usuários para outras páginas e *sites*, muitas vezes utilizando meios inseguros para determinar as páginas de destino. Sem devida validação, os atacantes podem redirecionar vítimas para *phishing* ou *malware*, ou usar o redirecionamento para acessar páginas não autorizadas.

Pela grande variedade de ataques, o mundo da segurança computacional é caracterizado pela evolução contínua, no qual novos ataques têm como resposta novas formas de proteção, que levam ao desenvolvimento de novas técnicas de ataques, de maneira que um ciclo é formado, (HOWARD, 2004). Nesse sentido, os sistemas computacionais não devem se preocupar somente com a adaptação ou construção de funcionalidades de defesa, para atender exclusivamente as ameaças explícitas, mas projetar uma arquitetura de *software* robusta para suportar vários tipos de ataques comuns e ainda os desconhecidos.

De acordo com Campos *et al.* (2006), um sistema computacional com objetivo de fortalecer a segurança da informação, controlando os riscos de sofrer ataques, deve conter as seguintes características:

- **Confidencialidade:** o conceito de confidencialidade é respeitado quando as informações são protegidas contra sua revelação, leitura e/ou cópia por alguém que não tenha sido explicitamente autorizado – interna ou externamente.
- **Integridade:** o princípio de integridade é respeitado quando a informação acessada está completa, sem alterações e, portanto, confiável.

---

<sup>54</sup> Documento eletrônico que comprova a identidade de uma pessoa, empresa ou *site* e garante a segurança das transações *online*.

- **Disponibilidade:** consiste na proteção dos serviços prestados pelo sistema de forma que eles não sejam degradados ou se tornem indisponíveis sem autorização, assegurando ao usuário o acesso aos dados sempre que deles precisar.
- **Autenticidade:** a autenticidade consiste em assegurar ao receptor que a mensagem é realmente procedente da origem informada em seu conteúdo. Geralmente é desenvolvida a partir de um mecanismo de senhas ou de assinatura digital.

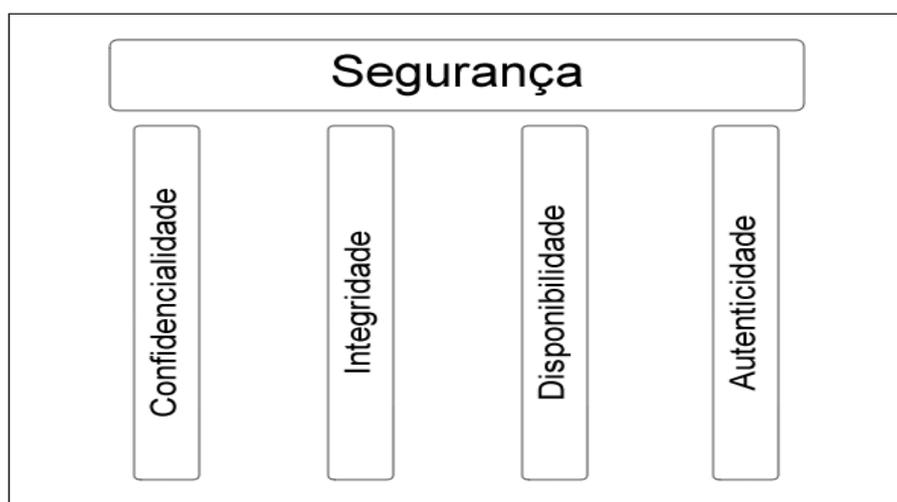


Figura 3 - Características da informação segura.

Na Figura 3, apresenta-se a base de um sistema de informação seguro. Se um desses princípios for desrespeitado em algum momento, isto significa uma quebra de segurança da informação, o que pode ser chamado de incidente de segurança da informação, (CAMPOS, 2006).

## 2.2 OWASP

OWASP - *The Open Web Application Security Project* é uma organização mundial sem fins lucrativos, dedicada a encontrar e combater as causas de insegurança em aplicações *Web*. Dentre vários projetos e ferramentas disponibilizadas pela OWASP, voltadas para segurança, encontra-se o Guia para Desenvolvimento Seguro de Aplicações *Web* e *Web Services* (OWASP, 2012). Este guia fornece orientações detalhadas para desenvolvedores e projetistas de *software* proporem soluções para o desenvolvimento seguro de aplicações *Web*.

Para o desenvolvimento de aplicações com bom potencial de segurança, ou seja, que toleram vários tipos de ataques presente na *Web*, é necessário a adoção de um conjunto de atividades, que auxiliem as organizações durante todo o ciclo de vida do *software*, a alcançar os objetivos de segurança.

De acordo WIESMANN *et al.* (2005), uma metodologia de desenvolvimento voltada para a segurança necessita dos seguintes atributos:

- Facilidade de adaptação para testes, *design* e documentação;
- Atividades onde os controles de segurança (tais como análise de risco, ameaças, revisões de código, etc.) podem ser aplicados;
- Métricas que auxiliam o aumento do nível de maturidade de segurança da organização e;
- Potencial para reduzir as taxas de erros correntes, e melhorar a produtividade do desenvolvimento.

Erros na construção de aplicações de *software* podem levar a falhas, e uma grande porcentagem destas se tornará vulnerabilidade de segurança (HOWARD, 2004). Assim, a maior meta é reduzir a chance dos desenvolvedores introduzirem vulnerabilidades de segurança, utilizando boas práticas de desenvolvimento, como a validação de entradas com o objetivo de verificar se os dados de entrada estão no padrão esperado pela

aplicação; e tratamento de exceções, para controlar alterações no fluxo normal da aplicação.

### **2.2.1 Modelagem de Riscos Estruturada a Ameaças**

A modelagem de riscos estruturada a ameaças é um método essencial para o desenvolvimento de *design*<sup>55</sup> seguro de aplicações *Web*, associadas aos controles de segurança, (AL-FEDAGHI *et al.*, 2010). Isto permite o direcionamento da organização e equipe de desenvolvimento, a utilizarem recursos, tempo e dinheiro com atividades relevantes e suficientes aos riscos reais.

Existem cinco etapas para a modelagem de riscos estruturada a ameaças: I) identificar os objetivos de segurança; II) obter a visão geral da aplicação; III) análise da aplicação; IV) identificação de ameaças e; V) identificação de vulnerabilidade (WIESMANN, 2005). Os passos são apresentados pela Figura 4.

---

<sup>55</sup> Desenho ou arquitetura do projeto de *software*, voltado para melhor entendimento e estruturação da solução proposta.

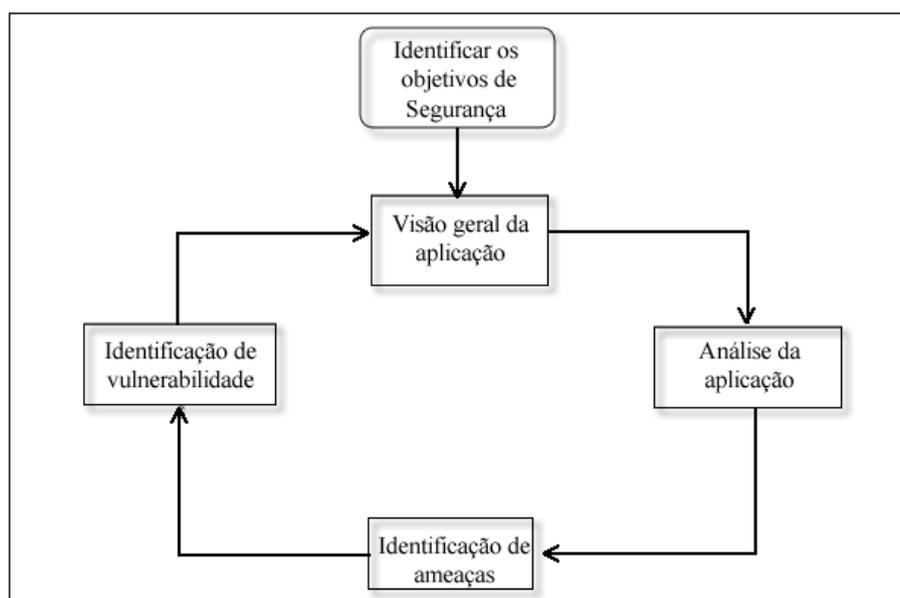


Figura 4 - Etapas para modelagem de riscos estruturada a ameaças. Adaptado de (WIEMANN, 2005).

### 2.2.1.1 Identificar os Objetivos de Segurança

Os objetivos de segurança são as intenções e restrições relacionadas com a confidencialidade, integridade, disponibilidade e autenticidade dos dados e da aplicação (MEIER, 2005). Assim, o ponto principal do objetivo de segurança é auxiliar no entendimento das atividades contidas em um ataque, com a finalidade de direcionar o desenvolvimento seguro nas áreas da aplicação que requerem mais atenção.

De acordo com (OWASP, 2007), para melhor identificar os objetivos de segurança de uma aplicação é relevante separá-los nas categorias de identificação, financeiro, reputação (prestígio), privacidade e regulamentação, e garantias de disponibilidades. A categoria de identificação tem o propósito de proteger a identidade dos usuários e disponibilizar informações somente aos usuários corretos. A categoria financeira apresenta

como objetivo avaliar se a organização está preparada para absorver uma potencial perda financeira. A terceira categoria (reputação) avalia qual o impacto negativo na imagem da aplicação, se um ataque a esta for bem sucedido. Privacidade e regulamentação analisam até que ponto a aplicação deve proteger os dados dos usuários, e se a aplicação está sujeita a se comportar de acordo com alguma regulamentação ou exigências da legislação. A categoria de garantias de disponibilidades verifica quais são os recursos presentes na aplicação *Web*, que realmente precisam de um alto nível de disponibilidade aos usuários. A identificação dos objetivos de segurança é importante, pois a aplicação correta dos controles de segurança irá economizar em recursos, tempo e dinheiro durante todo o ciclo de vida da aplicação.

#### **2.2.1.2 Visão Geral da Aplicação**

De acordo com Wiesmann (2005), uma vez que os objetivos de segurança tenham sido definidos, a visão geral da aplicação é obtida através da análise do *design* da aplicação *Web*, com a finalidade de identificar os componentes, os fluxos de dados e os limites de segurança. Neste sentido, com o objetivo de identificar as principais funcionalidades, características da aplicação e os clientes, Meier (2005) propõe as seguintes atividades:

- **Desenhar o cenário de implantação de ponta a ponta:** desenhar um diagrama simples que descreva a composição e estrutura da aplicação, seus subsistemas e suas características de implantação. Adicionar detalhes sobre a autenticação, autorização e mecanismo de comunicação;
- **Identificar os papéis:** identificar quais as atividades e funcionalidades que os usuários podem executar. A equipe envolvida no desenvolvimento do projeto necessita observar quais são os privilégios de um grupo de usuários; quais usuários podem ler e alterar dados; quais podem apagar os

dados; e varias outras atividades proporcionadas pela aplicação;

- **Identificar os principais casos de uso:** tem como objetivo ajudar a entender como o aplicativo será usado e como ele pode ser mal utilizado. Os casos de uso ajudam a identificar os fluxos de dados e as ameaças no processo de modelagem;
- **Identificar as tecnologias:** identificar as tecnologias como sistema operacional, banco de dados, linguagem de desenvolvimento e as tecnologias usadas para acessar a camadas da aplicação; ajuda a utilizar tecnologias específicas para tratamento de ameaças durante a modelagem da aplicação, e determinar as técnicas de mitigação correta e mais adequada;
- **Identificar os mecanismos de segurança da aplicação:** o objetivo dessa atividade é identificar como será realizada a validação dos dados de entrada, autenticação, o gerenciamento de configuração, tratamento dos dados sensíveis, gerenciamento de sessões, criptografia, manipulação dos parâmetros, tratamento de exceções e *login*.

### 2.2.1.3 Análise da Aplicação

De acordo com Meier, Mackman e Wastell (2005), nesta etapa realiza-se a decomposição da aplicação para reconhecer os limites de confiança, os fluxos de dados e os pontos de entrada e saída, de modo que um estudo mais profundo de identificação de características do aplicativo em desenvolvimento seja feito (OWASP, 2007):

- **Decomposição:** é adquirida através da identificação dos limites de confiança, para ajudar a focalizar a análise sobre as áreas inseguras;

- Identificação dos fluxos de dados: apresenta como finalidade entender como a aplicação interage com sistemas externos, e como os dados são validados e persistidos;
- Identificação dos pontos de entradas: Nessa etapa mapeiam-se os pontos que expõem as funcionalidades privilegiadas;
- Identificação dos pontos de saída: onde o aplicativo envia dados para o cliente ou sistemas externos, como banco de dados compartilhados.

#### **2.2.1.4 Identificação de Ameaças**

Para identificação de ameaças, Wiesmann, Curphey e Stirbei (2005) consideram os riscos conhecidos (CSRF, Falha de Injeção, XSS entre outros) para determinar quais implementações voltadas à segurança serão utilizadas para o desenvolvimento da aplicação. O mesmo autor ilustra a construção de um gráfico de ameaças apresentado pela Figura 5.

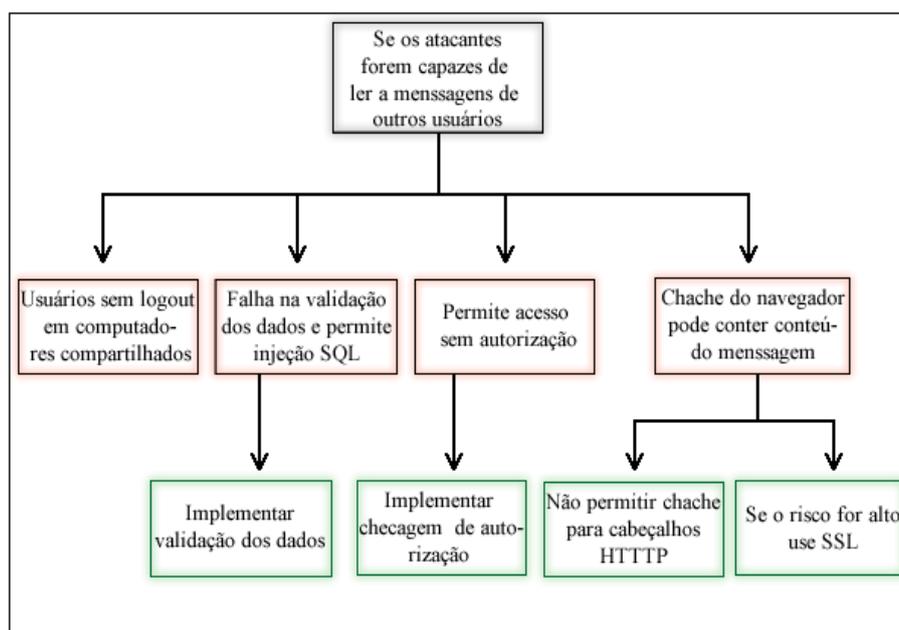


Figura 5 - Gráfico de identificação de ameaças. Adaptado de (WIESMANN, 2005).

Para conduzir o processo de identificação de ameaças, a equipe de desenvolvimento, arquitetos, profissionais de segurança, testadores e administradores do sistema devem trocar conhecimento para identificar as ameaças e ataques que possam afetar a aplicação e comprometer os objetivos de segurança.

### 2.2.1.5 Identificação de Vulnerabilidades

De acordo com Meier (2005), a identificação de vulnerabilidades baseia-se na revisão do quadro de segurança da aplicação *Web* a procura de vulnerabilidades. O mesmo autor afirma que as vulnerabilidades podem ser encontradas analisando:

- Camada por camada da aplicação, observando as funcionalidades de autenticação para realizar o mapeamento de quais conteúdos cada usuário pode obter acesso;

- Entrada e validação dos dados, com objetivo de verificar se os dados inseridos pelos usuários estão no formato esperado pela aplicação;
- Gerenciamento de configuração, para identificar se há o fornecimento de informações inseguras e garantir a configuração do servidor de aplicações, *frameworks* e servidor de bando de dados;
- Gerenciamento de sessão, para observar se as sessões podem ser criadas e mantidas durante toda a visita do usuário no sistema, de modo que os dados não sejam expostos;
- Criptografia dos dados, a fim de garantir a integridade, confidencialidade, disponibilidade e autenticidade da informação e;
- Sistema de *login*, com objetivo de encontrar falha na restrição das funcionalidades e conteúdos, aos usuários que utilizam o sistema.

### 2.3 Considerações Finais

Neste capítulo, foram apresentados os principais assuntos relacionados a aplicações *Web*, *Web 2.0*, segurança na *Web*, vulnerabilidades e ataques e os pilares da informação segura como autenticidade, disponibilidade, confidencialidade e integridade. A metodologia OWASP também foi abordada, com conceitos de modelagem de risco estruturada a ameaças e falhas em aplicações *Web*.

No Capítulo 3 será apresentado à metodologia do trabalho, a maneira de condução das atividades e o modo como os conceitos serão aplicados para alcançar o objetivo geral do trabalho.

### 3. METODOLOGIA

#### 3.1 Classificação da Pesquisa

Este trabalho realizou uma pesquisa aplicada, pois abordou a utilização de fatores de segurança em um sistema real. De acordo com Barros *et al.* (2000), a pesquisa aplicada apresenta como motivação a necessidade de produzir conhecimento para aplicação de seus resultados, com o objetivo de construir para fins práticos, uma solução para o problema encontrado na realidade.

Conforme afirmado por Avison (1999), a utilização da pesquisa-ação como forma metodológica possibilita a investigação de determinadas práticas de uma forma crítica e reflexiva. Sendo assim, o objetivo da pesquisa foi classificado como pesquisa-ação. À vista disto, foram considerados, em um sistema de comércio eletrônico (*e-commerce*), os problemas de segurança mais comuns presente na *Web* para planejar, desenvolver e avaliar um protótipo de *software*, de modo que as recomendações e técnicas apresentadas pela OWASP pudessem ser aplicadas e avaliadas.

De acordo com Wainer (2007), o método qualitativo se caracteriza por ser um estudo aprofundado de um sistema, onde se espera que o mesmo seja implantado, deste modo para atingir os objetivos do trabalho, foi adotada uma abordagem qualitativa. Contudo, utilizaram-se conceitos e recomendações de segurança para o desenvolvimento de uma aplicação *Web* segura. A aplicação *Web* desenvolvida, constituiu-se dos conceitos de um sistema de *e-commerce*, que apresenta as funcionalidades básicas para a realização segura de compras através da Internet.

## 3.2 Procedimentos metodológicos

Para consecução dos objetivos propostos neste trabalho, foram definidas atividades, conforme apresentado a seguir.

Inicialmente, foi realizada uma pesquisa bibliográfica em livros, guias e artigos buscando conceitos, recomendações e relatos sobre o desenvolvimento seguro de aplicações e sobre segurança na *Web*. À vista disto, utilizou-se uma metodologia que possibilitasse verificar a efetividade dos conceitos e recomendações, aplicadas no sistema de *e-commerce*, encontradas nas pesquisas.

Para tratamento dos riscos e ameaças presente no protótipo de *software (e-commerce)*, foi utilizada uma modelagem de riscos estruturada a ameaças, indicado por Wiesmann *et al.* (2005). Esta modelagem é composta por atividades para auxiliar no desenvolvimento de *design* seguro de aplicações *Web*, como conceituado na Sessão 2.2

## 3.3 Visão geral da aplicação

### 3.3.1 Descrição geral

A aplicação deve permitir que um usuário (cliente), navegue livremente pelo sistema, e adquira um ou mais produtos oferecidos, como ilustrado na Figura 6. De modo que para cada tipo de usuário foi determinada as restrições de acesso necessárias, como descrito abaixo:

- Administrador: este usuário possui acesso privilegiado sobre as funcionalidades do sistema. Incluindo o cadastramento, edição, listagem e remoção de produtos, clientes e administradores do sistema;

- Cliente: O usuário cliente está apto a comprar e pesquisar um ou mais produtos disponibilizados pelo sistema.

Mais detalhes e informações sobre as funcionalidades e o comportamento do sistema proposto pelo trabalho podem ser encontradas no Apêndice A.

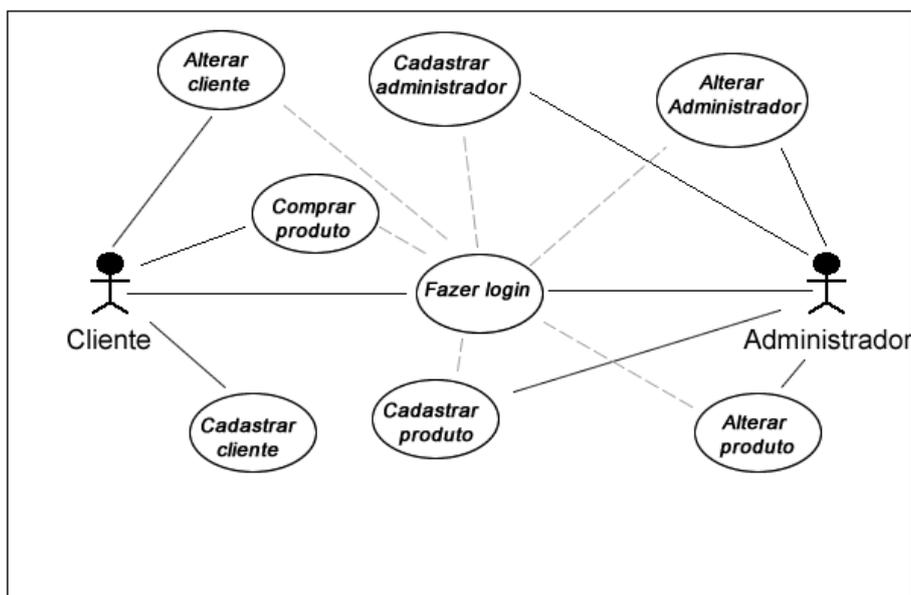


Figura 6 - Caso de uso para um sistema de *e-commerce* básico, onde as linhas tracejadas apresentam as ações do usuário que precisam fazer *login*.

### 3.3.2 Cenário

A aplicação desenvolvida apresenta as características de uma aplicação cliente - servidor como conceituado na Sessão 2.1.1 e exemplificado na Figura 7.

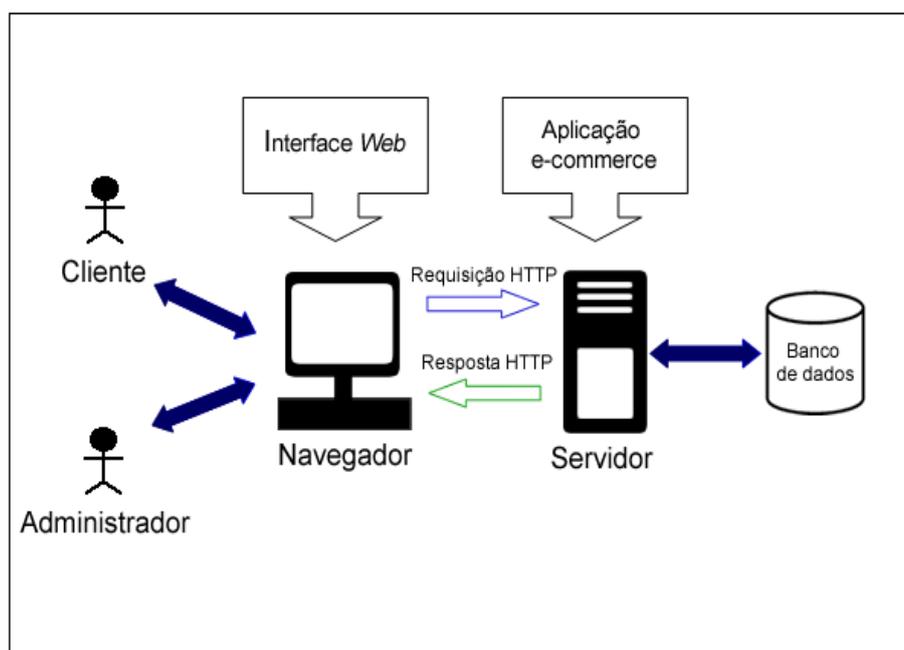


Figura 7 - Visão geral da aplicação de *e-commerce* desenvolvida pelo trabalho.

### 3.4 Objetivos de Segurança

As aplicações com comportamento de comércio eletrônico (*e-commerce*) proporcionam um conjunto relevante de facilidades para os usuários no processo de compra. Dentre este conjunto de facilidades está a possibilidade de não precisar sair de casa para adquirir um produto, e ter acesso a todas as informações sobre os produtos em uma pequena quantidade de tempo. No entanto, como na maioria das relações de negócios, o *e-commerce* também está sujeito a riscos e fraudes de segurança.

Para preservar a imagem da aplicação evitando que algum ataque seja bem sucedido, os objetivos de segurança definidos para o desenvolvimento do protótipo de *software* foram:

- Assegurar ao usuário que a comunicação realizada através do sistema é realmente procedente da origem informada em seu conteúdo;

- Proteger a identidade dos usuários e disponibilizar informações somente aos usuários corretos;
- Garantir que as informações acessadas estejam sem alterações, completas e, portanto confiável;
- Proteger os serviços prestados pelo sistema, para que não se tornem indisponível sem autorização, assegurando ao usuário o acesso aos dados sempre que deles precisar.

### **3.4.1 Restrições de segurança**

Para alcançar os objetivos gerais de segurança, descritos na Sessão 3.3, e a confidencialidade, integridade, autenticidade e disponibilidade dos dados na aplicação desenvolvida conceituada na Sessão 2.1.4, foram utilizados as seguintes restrições de segurança:

- Evitar que comandos ou consulta entrem no sistema como parte de dados enviados pelo usuário;
- Não enviar dados originados do usuário ao navegador, sem primeiramente validar ou codificar o conteúdo;
- Autenticar e gerenciar as sessões do usuário;
- Evitar expor uma referência a um objeto de implementação interna, como por exemplo, um arquivo, diretório, registro na base de dados, uma URL ou um parâmetro de um formulário;
- Evitar redirecionamento vulnerável de páginas;
- Criptografar dados sensíveis com algoritmos bem concebidos.

### 3.5 Tecnologias utilizadas

Após a definição de uma arquitetura do protótipo de *software*, foram utilizadas tecnologias voltadas para o desenvolvimento seguro de aplicações *Web* que suportassem os objetivos pretendidos por um sistema de compra e venda *online*.

#### 3.5.1 Linguagem de desenvolvimento

A linguagem utilizada para o desenvolvimento das funcionalidades básicas do sistema de comércio eletrônico foi a linguagem *Ruby*, juntamente com o *framework*<sup>56</sup> *Ruby on Rails*. *Ruby* é uma linguagem orientada a objetos, com tipagem forte, dinâmica e interpretada. Segundo Ruby (2012), uma das principais características de *Ruby* é a expressividade, pois se teve como objetivo desde o início que fosse uma linguagem muito simples de ler e ser entendida, para facilitar o desenvolvimento e manutenção de sistemas desenvolvidos com a mesma.

O *Ruby on Rails* foi criado para proporcionar maior agilidade, praticidade e segurança no desenvolvimento de aplicações *Web*, possuindo métodos auxiliares bem produtivos, como exemplo, contra Injeção de SQL (GUIDES, 2012).

A arquitetura de desenvolvimento em *Rails* é baseada no conceito de separar toda a aplicação em três camadas como: o modelo, visualização e controlador, ou seja, o MVC (*Model View Controller*). MVC é um padrão arquitetural onde os limites entre seus modelos, suas lógicas e suas visualizações são bem definidas, sendo mais simples fazer um reparo, uma mudança ou uma manutenção, já que essas três partes se comunicam de maneira desacoplada (THOMAS, 2005).

---

<sup>56</sup> Um conjunto de códigos comuns entre vários projetos de *software*.

### 3.5.2 Sistema operacional

O sistema operacional utilizado para desenvolvimento e teste da aplicação foi o Linux. O Linux foi criado em 1991 por Linus Torvalds, e hoje é mantido por uma comunidade mundial de desenvolvedores. O sistema adota a GPL, uma licença de *software* livre, proporcionando que todos os interessados podem usá-lo e redistribuí-lo, nos termos da licença (CAMPOS, 2012).

Segundo Silva (2008), com o objetivo de prover um ambiente pronto para a realização de diversas atividades desempenhadas por analistas de segurança e auditores, as distribuições Linux possuem como características: I) Coleta de Informação II) Mapeamento de Rede; II) Identificação de vulnerabilidade; IV) Penetração; V) Escalação de Privilégio VI) Controle de Acesso; VII) Análise de Rede de Rádio.

### 3.5.3 Banco de dados

O banco de dados utilizado para armazenamento dos dados presente do sistema de *e-commerce* proposto pelo trabalho foi o MySQL. O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (*Structured Query Language*) como meio de manipulação dos dados. É atualmente um dos bancos de dados mais populares, com mais de 10 milhões de instalações pelo mundo, possuindo as seguintes principais características (MYSQL, 2012):

- Portabilidade (suporta praticamente qualquer plataforma atual);
- Compatibilidade com várias linguagens de programação.
- Excelente desempenho e estabilidade;
- Pouco exigente quanto a recursos de hardware;
- Facilidade de uso;

- É um *software* livre com base na GPL;
- Replicação facilmente configurável;
- Meios gráficos de fácil utilização.

### 3.5.4 Mecanismos de segurança da aplicação

O Guia para Desenvolvimento Seguro de Aplicações *Web* e *Web Services* ressalta que, para obter uma aplicação *Web* segura, de acordo com Wiesmann (2005), é necessário atentar-se para validação dos dados de entrada, autenticação, gerenciamento de configuração, tratamento dos dados sensíveis, gerenciamento de sessões, criptografia, codificação de saída dos dados, tratamento de exceções e notificação de mensagens de erros detalhadas. Durante o desenvolvimento das funcionalidades básicas do sistema de *e-commerce*, foram aplicadas as recomendações e técnicas de segurança citadas para sanar as vulnerabilidades, garantir a disponibilidade dos dados, proteger informações e a identidade dos usuários.

Para que o protótipo de *software* contasse com mecanismos de segurança para a realização de compra segura através da Internet, considerando os principais ataques presente na *Web* como *Cross Site Request Forgery*, Falha de Injeção, Referência Insegura Direta a Objetos, *Cross-Site Scripting* (XSS), Falha de Autenticação e Gerenciamento de Sessão, Falha de Restrição de Acesso à URL, foram utilizadas recomendações e práticas de seguranças de acordo com o *The Ten Most Critical Web Application Security Risks* (OWASP, 2010), conforme apresentado a seguir.

- Validação de entrada: Como ilustrado na Figura 8, foi utilizado um mecanismo padrão de validação de entrada, antes dos dados serem exibidos ou armazenados na aplicação de *e-commerce*, para validar todas as entradas quanto ao tamanho, tipo, sintaxe e regras de negócio.

```

class Administrador < ActiveRecord::Base

  validates_presence_of :nome, :message => "deve ser preenchido"
  validates_presence_of :email, :message => "deve ser preenchido"
  validates_presence_of :password, :message => "deve ser preenchido"
  validates_presence_of :login, :message => "deve ser preenchido"

  validates_uniqueness_of :login, :message => " já existe. Favor alterar"
  validate :email_deve_ser_valido

  private
  def email_deve_ser_valido
    errors.add("email", "deve ser válido")
    unless email =~ /^[^@\s+]@((?:[-a-z0-9]+\.)+[a-z]{2,})$/i
    end
  end
end

```

Figura 8 - Exemplo de validação dos campos na entidade Administrador.

- Como apresentado pela Figura 9, foi especificado a codificação de saída (como UTF-8), para não permitir que o atacante tenha liberdade para determinar uma codificação de saída desejada.

```

module PO2
  class Application < Rails::Application

    # Configure the default encoding used in templates for Ruby 1.9.
    config.encoding = "utf-8"

  end
end

```

Figura 9 - Exemplo de codificação de saída com UTF-8.

- Evitou-se a notificação de mensagens de erros detalhadas que sejam úteis ao atacante, ou seja, mensagem que podem mostrar para o usuário mal intencionado qual dado está incorreto, como ilustrado pela Figura 10.

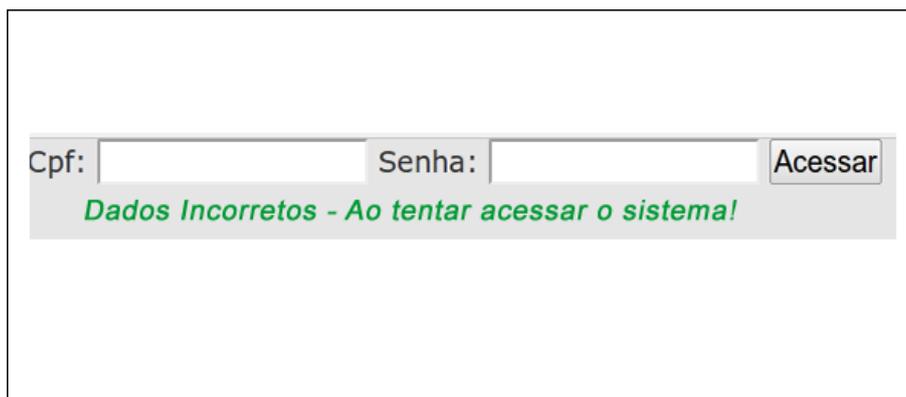


Figura 10 - Exemplo de mensagem segura emitida pelo sistema.

- Para criptografar os valores dos parâmetros sensíveis (CPF, senha e número do cartão de crédito) durante as requisições da aplicação, foi realizada uma filtragem de parâmetros como apresenta a Figura 11.

```
module P02
  class Application < Rails::Application
    | config.filter_parameters += [:password, :login, :cpf, :code, :number, :id]
  end
end
```

Figura 11 - Método que garante a não visualização dos dados presente nos parâmetros especificados.

- Foi verificado se os dados de entrada do usuário, enviados aos interpretadores da aplicação, não podiam modificar os comandos de manipulação do banco de dados. Como mostra o contorno em vermelho apresentado pela Figura 12.

```
def update
  @produto = Produto.find(params[:id])

  respond_to do |format|
    if @produto.update_attributes(params[:produto])
      format.html { redirect_to @produto, :notice => 'O produto foi alterado!' }
    else
      format.html { render :action => "edit" }
    end
  end
end
```

Figura 12 - Método que impede a passagem de comandos enviados pelos usuários.

- Evitou-se ações de ataque como *Cross Site Request Forgery* como conceituado na sessão 2.1.4, através da definição de um *token* de segurança como ilustrado pela Figura 13, calculado a partir da sessão atual, que será incluído em todos os formulários da aplicação.

```
class ApplicationController < ActionController::Base

  protect_from_forgery :secret => "123456789012345678901234567890"
```

Figura 13 - Definição do *token* de segurança, calculado a partir da sessão atual do usuário.

- Como apresentado pela Figura 14 e Figura 15, foi utilizado um mapeamento indireto de objetos, para não disponibilizar o diretório real dos arquivos da aplicação.

```
PO2::Application.routes.draw do

  #Configuração de mapeamento indireto - RESTful
  resources :produtos

end
```

Figura 14 - Mapeamento indireto de objetos - Configuração das rotas, no arquivo routes.rb, para acesso dos recursos da entidade Produto.

```
# GET /produtos
def index
  @produtos = Produto.all
end
# DELETE /produtos/1
def destroy
  @produto = Produto.find(params[:id])
  @produto.destroy
end
# POST /produtos
def create
  @produto = Produto.new(params[:produto])
end
```

Figura 15 - Mapeamento indireto de objetos - Modo como os recursos, da entidade Produto, foram disponíveis.

- Foi utilizado um mecanismo padrão para gerência de sessão, o objetivo desse gerenciamento é não expor os dados do usuário presente na sessão e determinar o prazo de expiração das sessões, como ilustrado pela Figura 16.

```
PO2::Application.config.session_store :cookie_store, {:key => '_PO2_session',
:secret => '0x0dkfj3927dkc7djdh36rkckdfzsg',
:expire_after => 2.minutes}
```

Figura 16 - Configuração de gerenciamento de sessão.

- Como ilustrado pela Figura 17, foi assegurado que todas as páginas apresentassem um *link* para o usuário sair do sistema, para evitar a permanência dos dados do usuário na sessão da aplicação.



Figura 17 - Botão para sair do sistema, presente em todas as páginas.

- Foi assegurado que os dados armazenados no disco fossem criptografados, como exemplifica a Figura 18.

```
def encrypt(password)
  encrypted = Digest::MD5.hexdigest(password)
  encrypted
end
```

Figura 18 - Método de codificação de senhas.

- Garantiu-se que todas URL's e funções de negócio fossem protegidas por um mecanismo de controle de acesso efetivo que verifique as funções e direitos do usuário antes que qualquer funcionalidade seja executada, como ilustrado pela Figura 19 e Figura 20.

```
class ApplicationController < ActionController::Base

  before_filter :require_login
  before_filter :require_ser_admin

  private
  def require_login
    if(session[:usuario_logged] == nil)
      flash[:notice] = "Você deve estar logado para realizar esta ação."
      redirect_to (:back) # Previne que a ação corrente seja executada
    end
  end

  def require_ser_admin
    if(session[:type] != "ADMIN")
      flash[:notice] = "Você não tem permissão para acessar."
      redirect_to root_url
    end
  end
end
```

Figura 19 - Interceptadores de acesso ao sistema e às funcionalidades destinadas ao Administrador.

```
class ClientesController < ApplicationController

  skip_before_filter :require_login, :only => [:new, :create]
  skip_before_filter :require_ser_admin, :only => [:new, :create]
```

Figura 20 - Filtro no Controlador na entidade Cliente para os recursos disponíveis e recursos destinados somente ao Administrador.

### 3.6 Considerações finais

Neste capítulo, foi detalhada a sequência das atividades realizadas no trabalho e o modo como as recomendações e técnicas de segurança proposta pela OWASP foram aplicados no sistema de *e-commerce* para alcançar o resultado esperado.

No Capítulo 4 serão detalhadas as funcionalidades e o comportamento do protótipo de *software* neste capítulo, apresentando as principais funcionalidades e como utilizá-las.

## 4. APLICAÇÃO DESENVOLVIDA

A aplicação *Web* projetada e desenvolvida para o presente trabalho constitui-se dos conceitos de um sistema de *e-commerce*, que apresenta as funcionalidades básicas para a realização de compra segura através da Internet.

### 4.1 Visão do usuário (cliente)

O sistema apresenta em suas páginas uma barra superior contendo do lado esquerdo um formulário para o usuário fazer o acesso ao sistema. No lado direito é apresentado dois *links*, um para criar uma conta na aplicação e outro *link* para o usuário sair da aplicação. No centro da página inicial são apresentados alguns produtos cadastrados no sistema, como apresentado pela Figura 21.



Figura 21 - Página inicial do sistema de comércio eletrônico.

Com a escolha de algum produto presente na página inicial, o usuário é direcionado para uma nova página com as principais características do produto escolhido, como preço, prazo de entrega, disponibilidade e classificação. Com o produto em destaque, o usuário pode adicionar o produto em seu carinho de compra e/ou finalizar a compra, como ilustrado pela Figura 22.



Figura 22 - Página para adicionar produto ao carinho de compras.

Se a escolha de finalizar a compra for realizada o usuário é direcionado para a página que mostra todos os produtos adicionados ao carinho de compra. Essa relação apresenta a descrição, preço, prazo de entrega, disponibilidade e classificação dos produtos adicionados ao carrinho de compra. Outras informações contidas nessa mesma página são as formas de pagamento a serem escolhidas pelo usuário, onde o sistema oferece a opção de pagamento com cartão, de modo que o usuário informe o código de segurança e o número do cartão para concluir a compra, como apresentado pela Figura 23.

**Produtos**

Valor Total: 1024.89

### Produtos adicionados no Carrinho

Descricao	Preco	Praco de entrega	Classificacao	Disponibilidade
Celular 1 3GS 8 GB	999.99	5 Dias	Celular	Disponível
Livro 1	24.90	4 Dias	Livro	Disponível

### Formas de Pagamento

**Cartão - à vista**



Código de segurança:

Número do Cartão:

**Boleto Bancário - à vista**



1 38055 65154 7

Figura 23 - Páginas dos produtos adicionados ao carinho de compra.

## 4.2 Visão do administrador

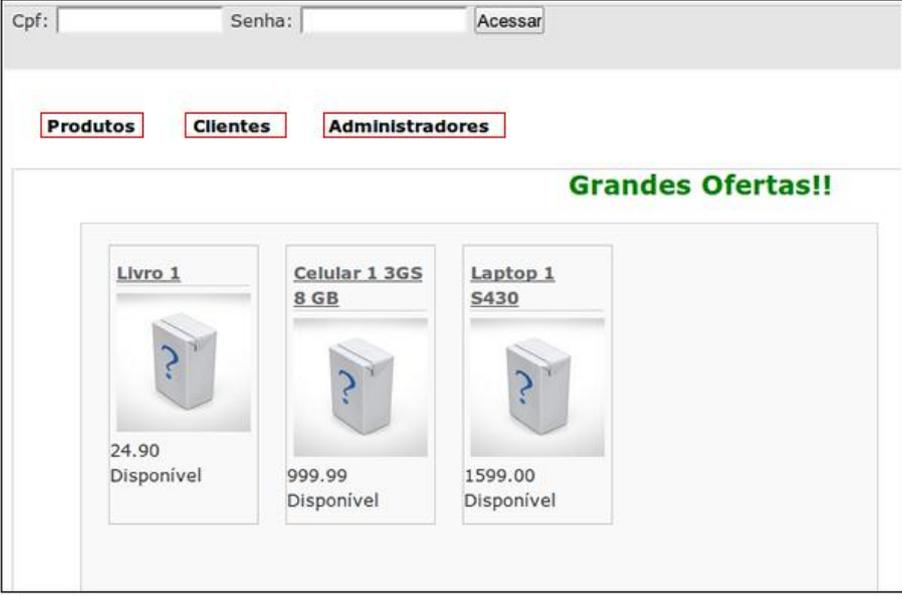
Na barra superior, presente em todas as páginas do sistema, é apresentado um *link* para o administrador fazer o acesso ao sistema. Clicando sobre esse *link*, o usuário é direcionado para uma página contendo um formulário. O formulário necessita como dados de entrada o *login* e senha de um administrador para obter acesso ao sistema, como ilustrado pela Figura 24.



The screenshot shows a web interface for administrator access. At the top, there are input fields for 'Cpf:' and 'Senha:' followed by an 'Acessar' button. In the top right corner, there are links for 'Criar Conta', 'Logout', and 'Adm'. The main heading is 'Acesso para Administrador'. Below this, there is a login form with 'Login:' and 'Senha:' fields, each enclosed in a red box, and an 'Acessar' button below them.

Figura 24 - Página para o Administrador acessar o sistema.

Como ilustrado pela Figura 25, quando um administrador acessa o sistema é disponibilizado três botões para esse tipo de usuário, sendo estes botões denominados como Produtos, Clientes e Administradores respectivamente.



The screenshot shows the administrator dashboard. At the top, there are input fields for 'Cpf:' and 'Senha:' followed by an 'Acessar' button. Below this, there are three buttons: 'Produtos', 'Clientes', and 'Administradores', each enclosed in a red box. The main heading is 'Grandes Ofertas!!'. Below this, there are three product cards, each with a placeholder image (a box with a question mark) and a price:

Produto	Preço	Status
Livro 1	24.90	Disponível
Celular 1 3GS 8 GB	999.99	Disponível
Laptop 1 S430	1599.00	Disponível

Figura 25 - Funcionalidades disponíveis ao Administrador.

Com acesso a sessão de Clientes, uma listagem de todos os clientes cadastrados ao sistema de *e-commerce* é disponibilizado ao administrador. Com essa listagem o administrador é capaz de remover, editar cada cliente e/ou adicionar um novo cliente na aplicação, como ilustrado pela Figura 26.



Nome	Email	Telefone			
wer	wer@com.br	12345	Mostrar	Editar	Remover
asdf	wer@com.br	12345	Mostrar	Editar	Remover
lucas	lucas@gmil.com	1234	Mostrar	Editar	Remover
tesdte	test@com.br	12345345	Mostrar	Editar	Remover
Lucas	lima4p@gmail.com	9259821	Mostrar	Editar	Remover

NOVO CLIENTE

Figura 26 - Listagem de todos os clientes cadastrados no sistema.

Com acesso a sessão de Administradores, o administrador tem uma listagem de todos os outros responsáveis pelo sistema de *e-commerce*. Na listagem obtida o administrador pode executar as funcionalidades de editar, remover cada responsável e/ou adicionar um novo responsável pela aplicação, como ilustrado pela Figura 27.

Nome	Login	Email			
ki	ki	ki	Mostrar	Editar	Remover
adf	adf	asdf@dot.br	Mostrar	Editar	Remover
asdf	asdf	asdf@dot.br	Mostrar	Editar	Remover
as	as	asdf@dot.br	Mostrar	Editar	Remover

NOVO ADMIN

Figura 27 - Listagem de todos os Administradores do sistema.

Com um clique na sessão de Produtos o administrador tem acesso às informações que um usuário (cliente) obtém e uma listagem geral de todos os produtos cadastrados no sistema. Com essa visão o administrador tem as opções de editar, remove cada produto e/ou adicionar um novo produto ao sistema, como apresentado pela Figura 28.

Produtos Clientes Administradores

**Grandes Ofertas!!**

**Livro 1**



24.90  
Disponível

**Celular 1 3GS 8 GB**



999.99  
Disponível

**Laptop 1 S430**



1599.00  
Disponível

**Queima de estoque**

**Queda de Preço Harry Potter**  
De últimas filmes da saga

a partir de R\$ **19.90**



**Computadores com Intel Core i3 + Monitor LED**

a partir de R\$ **1.200,00**



Codigo	Descricao	Preco	Prazo de entrega	Classificacao	Disponibilidade			
236	Livro 1	24.90	4 Dias	Livro	Disponível	Comprar	Editar	Remover
12	Celular 1 3GS 8 GB	999.99	5 Dias	Celular	Disponível	Comprar	Editar	Remover
8678	Laptop 1 S430	1599.00	6 Dias	Computadores	Disponível	Comprar	Editar	Remover

[Ver todos](#)

Figura 28 - Listagem de todos os produtos cadastrados no sistema.

Se a escolha de adicionar um novo produto ao sistema for realizada o administrador é direcionado para a página de cadastro de produtos. Para cadastrar um produto são necessários vários dados de entrada, como explicado no Apêndice A. Com a entrada de dados incorretos mensagens de validação são apresentadas ao administrador, como ilustrado pela Figura 29.

The screenshot displays a web interface with a navigation bar at the top containing the links "Produtos", "Clientes", and "Administradores". The main heading is "Cadastro de Produto". A prominent red error message box at the top left states: "Não foi possível salvar o Produto:" followed by a list of validation errors: "Preço deve ser preenchido", "Preço deve conter apenas números", "Classificação deve ser preenchido", "Prazo de entrega deve ser preenchido", and "Disponibilidade deve ser preenchido". Below the error box, the form fields are: "Codigo" (containing "365635"), "Descricao" (containing "Produto 04"), "Preço" (empty), "Prazo de entrega" (empty), "Classificação" (empty), and "Disponibilidade" (empty). A "Cadastrar" button is located at the bottom of the form.

Figura 29 - Formulário de cadastro de novos produtos com a ausência de alguns campos obrigatórios.

Com um clique no botão de remoção de um produto, uma nova janela de mensagem é lançada na tela, questionando o administrador se ele realmente está seguro em executar a ação de remoção do produto escolhido. Se a opção escolhida for sim, o produto é removido senão o produto será mantido no sistema, como apresenta a Figura 30.

Produtos    Clientes    Administradores

**Grandes Ofertas!!**

**Livro 1**



24.90  
Disponível

**Celular 1 3GS 8 GB**



999.99  
Disponível

**Laptop 1 S430**



1599.00  
Disponível

**Queima de estoque**

Queda de Preço  
**Harry Potter**  
Do Ministério da Magia

The page at localho:  
Você tem certeza?

Cancel    OK

+ Monitor LED  
a partir de R\$ 1.044,00

Figura 30 - Tentativa de remoção de algum produto no sistema.

Como ilustrado pela Figura 31, se a escolha de edição de um produto for realizada, o administrador tem acesso ao formulário de edição do produto escolhido com todas as informações já contidas no produto no valor dos campos de entrada. Nesse formulário o administrador pode ainda alterar os valores do produto escolhido.

Produtos   Clientes   Administradores

## Edição de Produto

Codigo  
236

Descricao  
Livro 1

Preco  
24.90

Praco de entrega  
4 Dias

Classificacao  
Livro

Disponibilidade  
Disponível

Cadastrar

[Mostrar](#) | [Voltar](#)

Figura 31 - Página para edição de Produto.

As funcionalidades apresentadas acima com as figuras de número 29, 30 e 31 também podem ser executadas para os Clientes e Administradores do sistema de *e-commerce*.

## 5. ANÁLISE E DISCUSSÃO

### 5.1 Considerações iniciais

Os materiais apresentados pela OWASP como o “Guia de Desenvolvimento Seguro de Aplicação *Web* e *Web Service*” (WIESMANN, 2005) e “*The Ten Most Critical Web Application Security Risks*” (OWASP, 2010), utilizados no trabalho, apresentam como características a objetividade, bons exemplos e ótima didática, como descrita a seguir.

- **Objetividade:** Os materiais são compostos por capítulos bem definidos que explicam e apresentam claramente os conceitos e atividades de segurança. Os conteúdos discutem e apresentam áreas-chaves de implementação como arquitetura, autenticação, gestão de sessões, controle de acesso e autorização, registro de eventos e validação dos dados.
- **Didática:** Todas as práticas de segurança apresentadas são compostas por métodos e técnicas que facilitam o entendimento dos ataques, que acontecem em aplicações *Web*, e ou questões a serem respondidas, por exemplo, para a identificação de vulnerabilidades.
- **Exemplos:** As ilustrações e os exemplos práticos demonstram os agentes, meios de ataques, vulnerabilidades e controles de segurança a serem utilizados. Desse modo, os exemplos apresentam como implementar, em projetos reais, as prevenções tais como validação de dados, gerenciamento de sessão, criptografia entre outros.

### 5.2 Análise da aplicação

A aplicação *Web*, desenvolvida pelo trabalho, constituída dos conceitos e funcionalidades básicas de um sistema de *e-commerce* foi testada

por Oliveira (2012). O objetivo dos testes foi ajudar na avaliação das técnicas e recomendações utilizadas, e verificar se os problemas de segurança considerados foram resolvidos.

Com a análise dos resultados obtidos pelo teste de segurança, foi possível observar que:

- As funcionalidades de autenticação, em cada camada da aplicação, realizam o mapeamento de quais conteúdos cada usuário pode obter acesso;
- O sistema não fornece informações de usuários válidos da aplicação que possibilitem a realização de ataques;
- Os dados inseridos pelos usuários estão no formato esperado pela aplicação;
- A aplicação faz uso de um *cookie* durante toda navegação utilizando valores criptografados;
- As sessões podem ser criadas e mantidas durante toda a visita do usuário no sistema, de modo que os dados não sejam expostos;
- Aplicação não realiza nenhum bloqueio, após certo número de tentativas erradas, no mecanismo de autenticação de *login* dos usuários;
- Sistema de *login* foi desenvolvido com eficiência, na restrição das funcionalidades e conteúdos, aos usuários que utilizam o sistema.
- Aplicação impossibilita o manuseio do *cookie* através de linguagens de *script*;
- Não houve verificação do tipo da entrada fornecida para executar funcionalidades como remover, editar e mostrar clientes do sistema.

Com o resultado dos testes observa-se que a modelagem de risco estruturada a ameaças, contida no guia da OWASP, foi essencial para configuração e implementação de código seguro no sistema de *e-commerce*. As atividades fornecem orientações detalhadas para projetistas e desenvolvedores de *software*, modelar e desenvolver uma aplicação atendendo as funcionalidades básicas, porém preocupando-se com os riscos de segurança mais graves que envolvem a aplicação *Web* em questão.

A identificação dos objetivos de segurança, para o sistema de *e-commerce*, possibilita a definição das atividades e prevenções necessárias para projetar e desenvolver uma aplicação segura. Com os objetivos de segurança definidos e com a aplicação dos controles de segurança, durante todo o ciclo de vida do sistema de *software*, foi possível:

- Preservar a identidade dos usuários e fornecer informações somente aos usuários corretos;
- Certificar ao usuário que os dados trocados através do sistema são consequentes da origem informada em seu conteúdo;
- Assegurar que as informações acessadas estejam completas, confiável e, portanto sem alterações.

Com a realização das recomendações e técnicas proposta por Wiesmann (2005), as pessoas envolvidas na construção do sistema são capazes de analisar o *design* da aplicação *Web* através da detecção dos principais usuários, funcionalidades e do cenário necessário para a execução do sistema. Com a realização desta análise, é possível identificar os fluxos de dados e seus componentes e caracterizar as funcionalidades da aplicação e seus usuários, ajudando a implementar as funcionalidades considerando os principais ataques presente na *Web* e as medidas de segurança para combatê-los.

### 5.3 Considerações finais

Neste capítulo, foi apresentada uma análise e discussão sobre a metodologia utilizada para a construção do protótipo de *software*, considerando os testes, conceitos e recomendações de segurança.

As falhas de segurança encontradas, tais como bloqueio de funcionalidades após certo número de tentativas erradas, e a falta de verificação do tipo da entrada fornecida para executar funcionalidades como remover, editar e mostrar clientes do sistema, não foram consideradas como vulnerabilidade, pois não foi possível violar a integridade do sistema a partir desta.

No Capítulo 6 será apresentada a conclusão das atividades e conceitos relatados no trabalho.

## 6. CONCLUSÃO

Considerando os objetivos levantados no início do trabalho, podemos concluir que foi desenvolvida uma aplicação *Web* com as funcionalidades básicas de um *e-commerce* seguro.

O uso de guias e metodologias que suportem os objetivos gerais de uma aplicação *Web*, em desenvolvimento, auxilia os envolvidos no projeto a construir uma aplicação segura. Nesse sentido, os princípios de segurança como confidencialidade, autenticidade, integridade e disponibilidade são valorizados durante todo o ciclo de vida da aplicação.

Com o aproveitamento das características dos materiais apresentados pela OWASP como objetividade, didática, bons exemplos, e a valorização dos princípios de segurança, o sistema de *e-commerce* proposto obteve uma arquitetura de *software* que resiste vários tipos de ataques comuns a sistemas *Web*. Essa característica de segurança da aplicação desenvolvida pode ser observada na análise dos resultados obtidos com os testes de segurança presente na Sessão 5.2.

As atividades de controle de segurança tais como análise de riscos, ameaças e revisão de código, proporcionaram facilidade na modificação dos códigos implementados e do *design* proposto. Desde modo, foi possível reduzir as taxas de erros correntes, e melhorar a produtividade do desenvolvimento. A chance de o desenvolvedor introduzir vulnerabilidades de segurança, utilizando de boas práticas de desenvolvimento, é consideravelmente baixa em relação ao desenvolvimento de projetos sem o auxílio de atividades voltadas para a segurança.

Contudo observa-se a importância de utilizar técnicas e recomendações de segurança em aplicações *Web*. No entanto é importante ressaltar que a conscientização, das pessoas envolvidas no projeto, a respeito de segurança e a utilização de tecnologias que auxiliem na execução de práticas de segurança, são extremamente importantes para que o projeto

alcance o objetivo principal que é garantia da integridade, confidencialidade, disponibilidade e autenticidade do sistema.

## 7. REFERÊNCIAS BIBLIOGRÁFICAS

AVISON, D. E., Lau, F., Myers, M. D., and Nielsen, P. A. (1999). **Action research**. Communications of the ACM, 42(1):94–97.

AL-FEDAGHI, S.; ALRASHED, A. **Threat Risk Modeling**. Kuwait. 2010.

BARROS, A. J. S. e LEHFELD, N. A. S. **Fundamentos de Metodologia: Um Guia para a Iniciação Científica**. 2 Ed. São Paulo: Makron Books, 2000.

CAMPOS, A. L. N. **Sistema de Segurança da Informação: controlando os riscos**. Florianópolis. 2006.

CAMPOS, A. **O que é o Linux**. Disponível em: <http://br-linux.org/faq-linux/> . Acessado em: 23/03/2012

CAMPI, M. **Brasil terá Centro de Defesa Cibernética**. 2011. Disponível em: <http://info.abril.com.br/noticias/ti/brasil-tera-centro-de-defesa-cibernetica-09062011-21.shl>. Acessado em 15/09/2011.

GUIDES; R. **Guia sobre segurança do Ruby on Rails**. Disponível em: <http://guias.rubyonrails.com.br/security.html>. Acessado em: 23/03/2012.

HOWARD, M. **Building More Secure Software with Improved Development Process**. McLean. 2004.

KUROSE, J. F. **Redes de Computadores e a Internet: uma abordagem top-down**. 5 ed. São Paulo. 2010.

LAWTON, G. **Web 2.0 Creates Security Challenges**. San Francisco. Pag-1. 2007.

MURUGESAN, S. **Understanding Web 2.0**. IT Professional. Pag-7. 2007.

MYSQL. **Why MySQL?** Disponível em: <http://www.mysql.com/why-mysql/>. Acessado em 23/03/2012.

MEIER, D. J.; MACKMAN, A.; WASTELL, B. **How to: Create a Threat Model for a Web Application at Design Time**. 2005 Disponível em: <http://msdn.microsoft.com/en-us/library/ms978527.aspx>. Acessado em 03/10/2011.

SILVA, G.; M. **Segurança em Sistemas Linux**. Rio de Janeiro.2008.

NIC.BR. **Estatísticas do CERT.br confirma tendência de ataques a servidores Web**. Disponível em: <http://www.nic.br/imprensa/releases/2011/rl-2011-19.pdf>. Acessado em: 24/08/2011.

OWASP. **Threat Modeling Risk**. 2007. Disponível em: [https://www.owasp.org/index.php/Threat\\_Risk\\_Modeling](https://www.owasp.org/index.php/Threat_Risk_Modeling). Acessado em 03/10/2011.

OWASP. **About The Open Web Application Security Project**. Disponível em: <https://www.owasp.org/index.php/OWASP>About>. Acessado em: 15/05/2012.

OLIVEIRA, T. T. **Testes de Segurança em Aplicações Web segundo a metodologia OWASP**. Lavras. 2012.

OWASP. **OWASP Top 10 - 2010 The Ten Most Critical Web Application Security Risks**. 2010.

RUBY, B. **O que é Ruby?** Disponível em: [http://ruby-br.org/?page\\_id=15](http://ruby-br.org/?page_id=15). Acessado em: 23/03/2012.

SCHNEIER, B. **Segurança.com Segredos e mentiras sobre a proteção na vida digital**. Rio de Janeiro. 2011.

SILVA, G.; M. **Segurança em Sistemas Linux**. Rio de Janeiro.2008.

THOMAS; D.; RUBY, S.; HANSSON, H. D. **Agile Web Development with Rails**. California. 2005.

UCHÔA, Q. J. **Introdução à Segurança Computacional**. Lavras. 2011.

WIESMANN, A.; STOCK, A.; CURPHEY, M; STIRBEI, R. **A Guide to Building Secure Web Applications and Web Services**. 2005.

WANG, F. **AI'S Hall of Fame: IEEE Computer Society**. Tucson. Pag-10. 2011.

WAINER, J. **Métodos de pesquisa quantitativa e qualitativa para a Ciência da Computação**. Sociedade Brasileira da Computação. 2007.

## 8. APÊNDICE A – REQUISITOS FUNCIONAIS DO *E-COMMERCE*

### 8.1.1 RF1 – Cadastro de administrador

**Descrição:** O Sistema deve permitir o cadastramento de um novo administrador. Este cadastro só poderá ser feito por outro administrador do sistema.

**Entrada:** Nome, *e-mail*, *login*, senha e perfil (administrador).

**Processamento:** O sistema fará a verificação se o *login* informado já existe. Caso o *login* exista será exibida a mensagem de alerta e solicitado que um novo *login* seja informado. Caso o *login* não exista será exibida a mensagem que o cadastro foi realizado com sucesso.

**Saída:** Mensagem: “O *login* já existe. Favor alterar e tentar novamente.” ou “O cadastro foi realizado com sucesso”.

### 8.1.2 RF2 – Alteração de administrador

**Descrição:** O Sistema deve permitir a alteração dos dados de um administrador. Esta alteração só poderá ser feita pelo próprio.

**Entrada:** Nome, *e-mail*, *login* e senha.

**Processamento:** O sistema fará a verificação se o *login* informado já existe. Caso o *login* exista será exibida a mensagem de alerta e solicitado que um novo *login* seja informado. Caso o *login* não exista será exibida a mensagem que o cadastro foi realizado com sucesso.

**Saída:** Mensagem: “O *login* já existe. Favor alterar e tentar novamente.” ou “O cadastro foi realizado com sucesso”.

### 8.1.3 RF3 – Cadastro de Produto

**Descrição:** O sistema deve permitir que o administrador cadastrasse os produtos no sistema.

**Entrada:** Código do produto, Descrição, Valor, Prazo de entrega, Formas de pagamento, Classificação, Disponibilidade.

**Processamento:** O sistema irá verificar se o código do produto informado já existe nos cadastros de produtos da empresa. Caso já exista, uma mensagem de alerta será exibida ou caso não exista, uma mensagem será exibida concluindo a operação.

**Saída:** Mensagem: “O produto informado já está cadastrado.” ou “Cadastro efetuado com sucesso”.

#### **8.1.4 RF4 – Alteração de Produto**

**Descrição:** O sistema deve permitir que o administrador pudesse alterar os dados de seus produtos oferecidos.

**Entrada:** Código do produto, Descrição, Valor, Prazo de entrega, Formas de pagamento disponíveis, Classificação, Disponibilidade.

**Processamento:** O sistema deve verificar se o código do produto alterado já existe nos cadastros de produtos da empresa. Caso já exista, uma mensagem de alerta será exibida ou se não existir, uma mensagem será exibida concluindo a operação.

**Saída:** Mensagem: “O produto informado já está cadastrado.” ou “Produto alterado com sucesso”.

#### **8.1.5 RF5 – Pesquisa de Produto**

**Descrição:** O sistema possibilitará a realização de pesquisas dos produtos cadastrados. Esta pesquisa poderá ser feita por qualquer usuário.

**Entrada:** Critério de pesquisa, Filtros por classificação e Ordenação.

**Processamento:** O sistema deve buscar os produtos conforme o critério de pesquisa e os filtros, ordenando-os sempre por ordem alfabética.

**Saída:** Lista na tela com os produtos pesquisados.

#### **8.1.6 RF6 – Compra de Produto**

**Descrição:** O sistema deve permitir a montagem de um pedido com os produtos selecionados por um usuário. Para finalizar a compra, o usuário deve ser cadastrado no sistema.

**Entrada:** Produtos selecionados (pedido), *login*, Senha, Forma de pagamento, Número de cartão e Senha do cartão ou Código de segurança.

**Processamento:** O sistema verificará a situação do cliente e listará as formas de pagamentos disponíveis para o pedido elaborado. Caso seja selecionada forma de pagamento cartão, será feita a chamada do sistema bancário, caso seja feita a forma de pagamento boleto será emitido o boleto. Ao finalizar a forma de pagamento, o sistema emitirá uma mensagem.

**Saída:** Chamada do sistema bancário ou imagem com o boleto e mensagem “Seu pedido foi efetuado com sucesso!”.

#### **8.1.7 RF7 – Cadastro de usuário (Cliente)**

-

**Descrição:** O sistema deve permitir que um novo cliente fosse cadastrado. Este cadastro é realizado pelo próprio usuário.

**Entrada:** Nome do cliente, E-mail, CPF, Endereço, Endereço de entrega, Telefone e Senha.

**Processamento:** O sistema irá validar o CPF digitado e verifica também se o mesmo já não possui cadastro. Caso o CPF seja inválido, o sistema solicita a correção. Caso o CPF já esteja cadastrado, será emitido o alerta de existente. Caso o CPF seja válido e não existente, finalizará o cadastro da cliente e emite uma confirmação .

**Saída:** Mensagem: “O CPF digitado não é válido.Favor alterar e tentar novamente” ou “O CPF digitado já existe. Favor alterar e tentar novamente” ou “Cadastro efetuado com sucesso. Receberá um e-mail de confirmação.”

#### **8.1.8 RF8 – Alteração de usuário (cliente)**

**Descrição:** O sistema deverá permitir que o cliente pudesse alterar seus dados cadastrais.

**Entrada:** Nome do cliente, E-mail, Endereço, Endereço de entrega, Telefone e Senha.

**Processamento:** Caso os dados sejam alterados, a alteração será concluída com a exibição de uma mensagem.

**Saída:** Mensagem: “Dados alterados com sucesso”.