1 INTRODUÇÃO

Segurança da Informação é a área do conhecimento que protege os ativos de informação contra acessos não autorizados, alterações indevidas ou, ainda, a sua indisponibilidade (SÊMOLA, 2003). Trata-se, portanto, de uma área que envolve conhecimentos de criptografia, integridade e disponibilidade.

O presente trabalho enquadra-se no tema de Segurança da Informação. Mais especificamente, será dado enfoque aos sistemas corporativos do Exército Brasileiro e os desafios de segurança que estes enfrentam, atualmente, com relação aos conceitos citados. Em outras palavras, este trabalho versa sobre a Segurança da Informação nos Sistemas Corporativos do Exército Brasileiro.

1.1 Problema da Pesquisa

Até recentemente, os ataques a sistemas informatizados concentravam-se nas redes de computadores. Os sistemas corporativos que utilizam a rede não eram alvos comuns dos *hackers*. Atualmente, esse cenário mudou. Os anos de esforços em desenvolvimento de mecanismos de proteção para as redes, como *Firewalls*, Anti-Vírus, Sistemas de Detecção de Intrusão (IDS - *Intrusion Detection System*), entre outros, resultaram em redes mais seguras e com poucas vulnerabilidades a investidas externas. Assim, com o aumento de segurança das redes, tornou-se mais fácil atacar o sistema computacional em si, ou seja, os aplicativos que compõem o sistema informatizado que rodam na rede. De acordo com a ISC² (2009), a maior parte das falhas de segurança estão relacionadas aos aplicativos, "*Some 80% of all security breaches are application related. Every person involved in the Software Lifecycle should consider security as an essential element.*". Portanto, verifica-se que o foco dos ataques, atualmente, mudou das redes para os *softwares*. Assim, verifica-se que o ponto crítico de segurança da informação hoje são os sistemas corporativos, e não mais as redes de

computadores.

O Exército Brasileiro, cuja missão constitucional primordial é a defesa da Pátria (BRASIL, 1988), possui diversos sistemas corporativos que o auxiliam a cumprir essa missão. E, como qualquer corporação de grande porte, o EB também depende dos seus sistemas de informação para executar as suas tarefas diárias. Assim, uma falha de segurança em um sistema corporativo do Exército pode prejudicar o cumprimento da sua missão constitucional de defender a Nação. "A visão corporativa da segurança da informação deve ser comparada a uma corrente, em que o elo mais fraco determina seu grau de resistência e proteção. A invasão ocorre onde a segurança falha!" (SÊMOLA, 2003, p. 40). Portanto, não adianta ter uma rede corporativa forte e bem protegida se os sistemas corporativos que rodam nessa rede são fracos em termos de segurança e vulneráveis a ataques; a invasão ocorrerá no elo mais fraco do conjunto rede e aplicativos que, atualmente, são justamente os sistemas corporativos.

Dessa forma, tendo em vista que a Segurança da Informação é indispensável para o Exército Brasileiro e que, na atual conjuntura da Tecnologia da Informação, os ataques virtuais concentram-se nos programas e não mais na rede em si (PAUL, 2009), verifica-se a importância de estabelecer-se um conjunto de procedimentos de Segurança da Informação para os sistemas corporativos do EB. Este é, justamente, o problema alvejado pelo presente trabalho.

Em outras palavras, o problema abordado resume-se em responder a seguinte pergunta: quais são os procedimentos (ou boas práticas) a serem adotadas nos sistemas corporativos do Exército Brasileiro, a fim de obter-se um bom nível de segurança?

1.2 Hipóteses de Investigação

Na implantação de procedimentos (boas práticas) de segurança da informação nos sistemas corporativos do Exército Brasileiro, assume-se como hipóteses:

- garantia de propriedades desejáveis, como confidencialidade e integridade dos dados;
- eliminação de vulnerabilidades do sistema corporativo;
- redução dos riscos de segurança.

1.3 Justificativa

Com a modernização do Exército Brasileiro e o avanço da Tecnologia da Informação, os processos e sistemas do EB estão sendo informatizados. A automação dos processos resulta em maior agilidade e eficiência geral do sistema. Entretanto, o ambiente virtual dos computadores e de suas redes de comunicação (Internet e LANs) expõe os sistemas a novos tipos de riscos e ataques de segurança. Para mitigar esses riscos torna-se necessário o desenvolvimento de procedimentos (ou de uma cultura) de renomadas boas práticas de segurança digital.

Tendo em vista que o problema de segurança da informação durante o desenvolvimento de sistemas corporativos é uma preocupação recente, é necessário dedicar esforços para tornar a camada de aplicações mais seguras. O desenvolvimento de um trabalho de conclusão de curso desta natureza contribui para criar uma cultura de segurança da informação no desenvolvimento de sistemas para o Exército; ou seja, ajuda a tornar os desenvolvedores e usuários dos sistemas corporativos do EB conscientes com relação aos cuidados que devem ser tomados com os sistemas de informação. Com uma cultura de segurança estabelecida, o ambiente de trabalho virtual do Exército torna-se mais seguro em termos de Tecnologia da Informação. Assim, é possível evitar problemas relacionados a segurança dos dados, como perda de informações, exposição de textos confidenciais e falhas de integridade.

O Exército Brasileiro, como instituição, e os usuários dos seus sistemas corporativos serão beneficiados pela adoção dessas práticas de Segurança da Informação. Os usuários têm a garantia de integridade dos seus dados e ganham com a maior eficiência do sistema informatizado; O Exército, por sua vez, será beneficiado pela redução dos riscos de segurança em seus sistemas corporativos.

1.4 Objetivos

O presente trabalho tem por objetivo definir boas práticas na área de Segurança da Informação para o desenvolvimento de sistemas corporativos do Exército Brasileiro.

Adicionalmente, a fim de atingir-se o referido objetivo geral, tem-se como objetivos específicos:

- Realizar o levantamento bibliográfico dos conceitos teóricos;
- Propor procedimentos (boas práticas) de segurança a serem adotados;
- Validar a proposta feita no trabalho através mensuração dos níveis de segurança antes e depois da aplicações dos procedimentos propostos.

1.5 Metodologia

A metodologia utilizada consiste, inicialmente, da revisão bibliográfica dos conceitos teóricos, o desenvolvimento dos procedimentos propostos de Segurança da Informação e, por fim, a validação das propostas de segurança.

A revisão bibliográfica da teoria e dos principais conceitos de Segurança da Informação aborda tanto assuntos de ordem técnica (métodos de criptografia, assinatura digital, autenticação, entre outros) como itens de ordem processual (riscos e engenharia social, por exemplo). A pesquisa bibliográfica fundamenta-se em documentação indireta, consistindo de consultas a fontes primárias (normas técnicas ABNT e ISO) e, também, a fontes secundárias (livros e artigos).

O desenvolvimento dos procedimentos de Segurança da Informação foi realizado com base em renomadas boas práticas, amplamente reconhecidas e, principalmente, fundamentadas em bibliografia sólida. Os procedimentos englobam tanto itens técnicos como processuais. Dessa forma, os procedimentos propostos podem nortear o comportamento dos sistemas, dos seus usuários e dos desenvolvedores.

1.6 Organização do Trabalho

Este texto está organizado conforme descrito a seguir. No capítulo 2, Segurança da Informação, apresenta o ciclo de vida de desenvolvimento de *software* e como a segurança é

inserida nesse ciclo. A seguir, são analisados os conceitos fundamentais da área de Segurança da Informação. Também é apresentada a classificação do Governo Federal para as informações segundo o seu grau de sigilo. Esses conceitos são essenciais para o entendimento da proposta feita por este trabalho e, também, são de conhecimento indispensável a qualquer profissional que trabalhe com segurança em sistemas de computação.

No capítulo 3, Procedimentos de Segurança, é apresentada a proposta de procedimentos de Segurança da Informação para os sistemas corporativos do Exército Brasileiro, que é composta por quatro itens principais: treinamento do pessoal, revisão do código-fonte, verificação automatizada de vulnerabilidades no código e, por fim, casos de teste de segurança e vulnerabilidade.. O capítulo 4, Validação, mostra como validar a proposta apresentada, com os procedimentos a serem adotados e o cronograma de atividades a ser seguido. Por fim, no capítulo 5, será feita uma conclusão.

2 SEGURANÇA DA INFORMAÇÃO

Este capítulo apresenta os conceitos básicos de Segurança da Informação. A ciência e compreensão desses conceitos são indispensáveis para o entendimentos e a correta aplicação dos procedimentos de segurança propostos neste trabalho. "Conceitos sólidos e seu claro entendimento são a matéria-prima que implicará na qualidade e no resultado dos trabalhos" (SÊMOLA, 2003, p. 75). Por isso, esses elementos fundamentais serão explicados em detalhes a seguir.

O primeiro conceito a ser analisado é o de ciclo de vida no desenvolvimento de *software* e como a segurança da informação deve ser integrada nesse ciclo. A seguir é apresentado o conceito mais importante em termos de segurança, que é o de *Confidentiality Integrity Availability Authentication Authorization Auditing* (CIA + A³). Uma noção de criptografia também é apresentada, com abordagem tanto de métodos simétricos como assimétricos. Finalmente, são apresentadas as classes de sigilo do Governo Federal.

2.1 Segurança no SDLC

O processo de desenvolvimento de software é conhecido como *Software Developement Life Cycle* (SDLC), ou Ciclo de Vida do Desenvolvimento de *Software* (CVDS). "O processo de desenvolvimento de *software* é um conjunto de termos nos quais pode-se descrever e discutir diferentes metodologias (...) que variam muito na sua implementação, mas que compartilham fases em comum de uma forma ou de outra. Todas essas fases estão sob o conceito de *Software Development Lifecycle (SDLC)*". (ARNOLD, et. al, 2007, p. 1). As fases a que os autores citados referem-se são: planejamento, desenvolvimento e teste. Outros autores incluem ainda, no ciclo de vida, as fases de definição de requerimentos e manutenção. (KOCK, 2006, p. 5). Assim, pode-se verificar, pela literatura especializada, que as fases do ciclo de vida do desenvolvimento de *software* variam muito,

especialmente quanto à nomenclatura de cada fase e a quantidade total de fases. Já sobre as referidas metodologias, ou modelos, "Existem diversos modelos de ciclo de vida de desenvolvimento de *software* em uso atualmente. *Waterfall, spiral, rapid application design, joint application design* e *prototyping* são os cinco modelos mais utilizados por programadores e engenheiros de *software* ao desenvolver projetos" (TIPTON, KRAUSE, 2007, p. 242).

Uma análise detalhada de cada uma dessas fases e metodologias foge do escopo deste trabalho, tendo em vista que o foco de interesse é a segurança no desenvolvimento de um projeto. Entretanto, a definição de cada uma das fases é importante, pois os controles de segurança propostos neste trabalho far-se-ão presentes em todas as etapas do ciclo de vida do *software*. O ciclo de vida adotado neste trabalho é mostrado na figura 2.1. Este modelo em particular foi elaborado tendo em vista que o autor possui familiaridade e experiência prévia com o *Capability Maturity Model Integration* (CMMI, 2009).

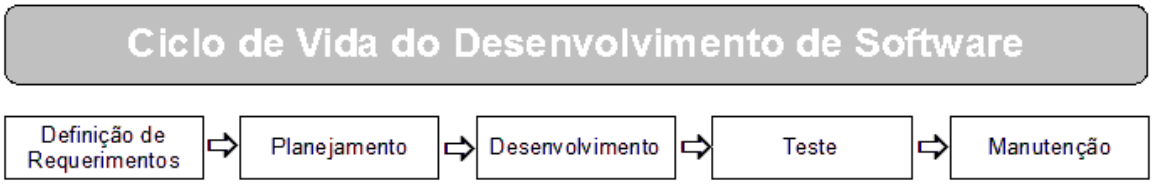

Figura 2.1: Ciclo de Vida do Desenvolvimento de *Software*

Fonte: elaborado pelo autor

Analisando a figura apresentada, verifica-se que o ciclo de desenvolvimento de *software* possui fases distintas para a definição de requisitos, para o planejamento das atividades seguintes, assim como distingue as fases de codificação do *software*, da série de testes pelo qual o sistema será submetido e, finalmente, a fase de manutenção do programa, que perdurará até o final da sua vida útil. Neste trabalho serão apresentados controles específicos de segurança a serem inseridos em cada uma dessas fases do ciclo de vida de desenvolvimento de um *software*.

O mais importante a ressaltar nesse momento é sobre o custo de correção de *bugs* durante o ciclo de vida do desenvolvimento de um programa. O custo de resolver defeitos de

software aumenta exponencialmente à medida que o produto progride através do ciclo de vida do *software*, "*It is also important to note that the cost of resolving software defects increases exponentially as the product progresses through the software lifecycle*" (FARIS, 2006, p. 30). Em outras palavras, quanto mais tarde uma falha for resolvida, mais caro será o custo da sua resolução. A figura 2.2 ilustra o aumento desse custo.

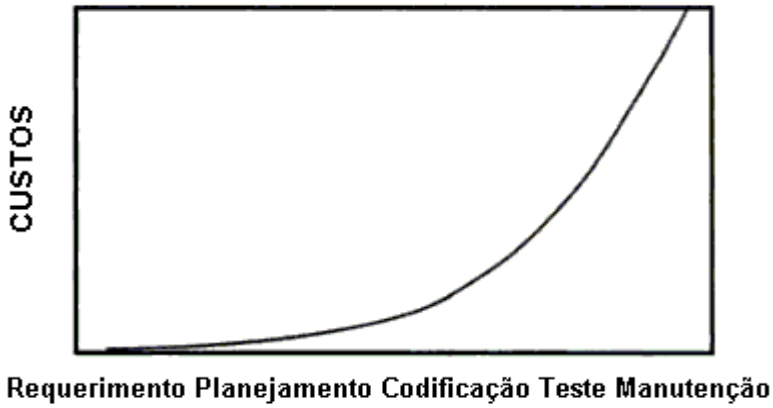

Figura 2.2: custo de resolução de defeitos ao longo do ciclo de vida do *software*
Fonte: elaborado pelo autor

Portanto, independente de qual for o modelo de SDLC utilizado, o importante é que a segurança deve ser parte intrínseca do projeto, em todas as suas fases. Dessa forma, evitar-se-á custos na correção de falhas, ao mesmo tempo que o projeto será inerentemente seguro.

2.2 Conceitos Fundamentais

A Segurança da Informação, apesar do nome moderno, é uma preocupação muito antiga. Na Roma Antiga, o imperador Júlio César já se preocupava com o problema, enviando mensagens cifradas para os seus generais, a fim de preservar a confidencialidade de suas ordens. O código utilizado para criptografar as mensagens ficou conhecido como a Cifra de César e, atualmente, é reconhecido como um dos mais antigos algoritmos de criptografia de

que se tem registro histórico (TANENBAUM, 2003).

Um dos conceitos mais bem difundidos em termos de boas práticas em segurança da informação é a dupla conhecida como CIA + A³ ou CIA AAA (leia-se "*CIA Triple-A*"). As siglas, em inglês, significam: *Confidentiality Integrity Availability, Authentication Authorization and Auditing*. Trata-se de um conjunto de conceitos importantíssimos de segurança da informação que, traduzidos para o português, significam: Confidencialidade Integridade Disponibilidade, Autenticação Autorização e Auditoria.

Confidencialidade é a capacidade de proteger a informação de tal forma que ela possa ser acessada e compreendida apenas pela a pessoa ou entidade a qual se destina. A confidencialidade previne o acesso não autorizado aos dados. Uma das técnicas mais utilizadas para a obtenção da confidencialidade é o uso da criptografia (SCHNEIER, 2007). Um algoritmo criptográfico embaralha a informação de tal forma que ela torna-se ilegível. Esse processo, obviamente, é reversível mediante a apresentação de uma chave. No caso do algoritmo criptográfico trabalhar com uma única chave, ele é chamado de algoritmo simétrico. Por outro lado, se o algoritmo realizar a sua computação com duas chaves distintas, o algoritmo é chamado de assimétrico. Existem outras técnicas, além do uso de criptografia, de obter-se confidencialidade. A mais simples de todas é simplesmente omitir ou ocultar a informação sigilosa, exibindo somente parte da informação como um todo. Uma outra maneira de fornecer confidencialidade para informações sensível é protegendo o acesso ao meio no qual a informação se encontra.

A **integridade** é a propriedade de averiguar se um determinado conjunto de informações foi alterado ou não (STEWART, et. al., 2005). Por exemplo, garantir a integridade de um arquivo de texto no sistema de arquivos de um computador significa ter a capacidade de determinar, com certeza, se o arquivo foi alterado ou não. Em um sistema corporativo, a integridade reflete-se em garantir que nenhuma informação do sistema, seja ele em parte ou como um todo, foi alterada; caso ocorra alguma alteração, o sistema deve ser capaz de detectar a alteração automaticamente. Salienta-se que a alteração das informações pode ser tanto acidental como intencional. A integridade garante que os dados não foram alterados durante o trânsito.

Disponibilidade significa ter a informação disponível quando ela for requisitada. "*Availability also implies that the supporting infrastrcutre (...) is functional and allows authorized users to gain authorized access"* (STEWART, et. al., 2005, p. 156), isto é, a

disponibilidade também implica que a infra-estrutura de suporte esteja funcional e permita que usuários autorizados tenham acesso. A infra-estrutura a que o autor referenciado se refere inclui servidores de rede, os meios de comunicações, assim como os mecanismos de controle de acesso.

Autenticação é a capacidade de identificar quem é o usuário, ou seja, averiguar se trata-se da pessoa A ou B. Esse processo de identificação pode ser realizado de diversas formas. De um modo geral, pode-se analisar as seguintes propriedades de um usuário para realizar a sua autenticação: o que o usuário **sabe**, o que ele **tem** e o que ele **é** (STEWART, 2005). Ao analisar-se exemplos de métodos de autenticação, verifica-se que são essencialmente essas propriedades do usuário que estão sendo verificadas. Assim, pode-se averiguar que, por exemplo, com o emprego de usuário e senha, está-se analisando o que o usuário sabe; já com o uso de um cartão magnético, verifica-se o que o usuário tem; com a análise da impressão digital ou da retina ocular, a autenticação certifica o que o usuário é, ou seja, verifica alguma de suas propriedades fisiológicas. A autenticidade garante que os dados são de autoria do remetente. E a **não-repudiação** garante que o remetente não pode repudiar a autoria de uma mensagem sua (WINDLEY, 2005).

Uma vez autenticado, isto é, estabelecido que o usuário é realmente quem ele alega ser, o passo seguinte é determinar o que ele pode fazer. **Autorização** é, portanto, uma lista de ações ou comandos que o usuário pode executar no sistema (STAMP, 2005). Por exemplo, a maioria dos sistemas operacionais divide os seus usuários em, no mínimo, dois grupos: um grupo de administradores e outro de usuários comuns. O primeiro é mais restrito e os seus usuários podem realizar mais ações, como gerenciar o sistema, criar novas contas, instalar novos programas, entre outras tarefas; já o segundo grupo, que constitui a grande maioria dos usuários no exemplo de sistema operacional, possui acesso mais limitado, podendo apenas executar as tarefas mais comuns. Com esse exemplo, ilustra-se a utilidade do conceito de autorização: segregar os usuários do sistema conforme a sua função.

O último conceito da tríade AAA é o de *Auditing* ou, traduzindo para o português, ter-se-ia "ato de fazer auditoria" ou, simplesmente, **Auditoria**. Este conceito também é chamado por alguns autores de *Accounting* ou Rastreabilidade, em português. Conforme o nome do termo traduzido sugere, esse conceito traduz-se na capacidade do sistema corporativo rastrear as ações dos usuários, o consumo de recursos, e as ações de gerenciamento do sistema (RUSSEL, GANGEMI, 1991). Tal rastreamento pode ser implementado em arquivos de *log*

(registro) ou em banco de dados. São essas informações que permitem a realização de auditorias sobre o sistema.

Ativo é um elemento de valor para uma organização ou um indivíduo e que, por isso, precisa de proteção adequada (NBR/ISO/IEC-17799). A própria informação e os processos que a manipulam são considerados ativos, bem como os equipamentos em que a informação é manipulada, armazenada, transportada e descartada (SÊMOLA, 2003).

Atualmente, com o crescente desenvolvimento dos sistemas corporativos de computação, pode-se dizer que a informação é um ativo cada vez mais valorizado. "A Informação representa a inteligência competitiva dos negócios e é reconhecida como ativo crítico para a continuidade operacional e saúde da empresa" (SÊMOLA, 2003, p. 39).

Por fim, é importante definir o conceito de ***Hash***, que frequentemente é confundido com a criptografia em virtude de algumas semelhanças que apresentam. Ambos são funções que convertem dados. A diferença é que no *hashing* não é possível obter os dados originais de volta, enquanto na criptografia é possível tanto cifrar, como decifrar os dados.

2.3 Classes de Sigilo do Governo Federal

O Decreto número 4.553, de 27 de dezembro de 2002, dispõe sobre a salvaguarda de dados, informações, documentos e materiais sigilosos de interesse da segurança da sociedade e do Estado, no âmbito da Administração Pública Federal. São considerados sigilosos "dados ou informações cujo conhecimento irrestrito ou divulgação possa acarretar qualquer risco à segurança da sociedade e do Estado, bem como aqueles necessários ao resguardo da inviolabilidade da intimidade da vida privada, da honra e da imagem das pessoas". O decreto salienta, ainda, que o acesso a dados ou informações sigilosos é restrito e condicionado à necessidade de conhecer.

O conhecimento da classificação de sigilo das informações é indispensável ao desenvolvedor de *software* para que este possa implementar os controles de segurança necessários à manutenção da confidencialidade dos dados e, também, cumpra as exigências impostas pela legislação. Por isso, são transcritas, a seguir, as definições de informações ou dados sigilosos que, em razão do seu teor ou dos seus elementos intrínsecos, são classificados

em ultra-secretos, secretos, confidenciais e reservados; tudo conforme o artigo 5º do Decreto número 4.553, de 27 de dezembro de 2002:

- ultra-secreto: "dados ou informações referentes à soberania e à integridade territorial nacionais, a planos e operações militares, às relações internacionais do País, a projetos de pesquisa e desenvolvimento científico e tecnológico de interesse da defesa nacional e a programas econômicos, cujo conhecimento não-autorizado possa acarretar dano excepcionalmente grave à segurança da sociedade e do Estado".

- secreto: "dados ou informações referentes a sistemas, instalações, programas, projetos, planos ou operações de interesse da defesa nacional, a assuntos diplomáticos e de inteligência e a planos ou detalhes, programas ou instalações estratégicos, cujo conhecimento não-autorizado possa acarretar dano grave à segurança da sociedade e do Estado".

- confidencial: "dados ou informações que, no interesse do Poder Executivo e das partes, devam ser de conhecimento restrito e cuja revelação não-autorizada possa frustrar seus objetivos ou acarretar dano à segurança da sociedade e do Estado".

- reservado: "dados ou informações cuja revelação não-autorizada possa comprometer planos, operações ou objetivos neles previstos ou referidos".

Cabe destacar ainda que, em seu artigo 7º, o referido decreto ainda estabelece os prazos de duração da classificação anteriormente transcrita. Os prazos vigoram a partir da data de produção do dado ou informação, e são conforme a seguir:

I ultra-secreto: máximo de cinqüenta anos;
II secreto: máximo de trinta anos;
III confidencial: máximo de vinte anos; e
IV reservado: máximo de dez anos.

Os prazos de duração podem ser renovados uma única vez, no máximo pelo período definido no caput, com a exceção dos documentos ultra-secretos, que podem ser renovados indefinidamente.

3 PROCEDIMENTOS DE SEGURANÇA

Segundo Sêmola (2003), segurança consiste em implementar controles que reduzam o risco a níveis adequados, viáveis e administráveis. Os procedimentos de segurança propostos neste capítulo constituem, nas palavras do autor citado, um controle; este controle em particular, objetiva a redução do risco de falhas de segurança durante o desenvolvimento de *software*. Ou seja, esses procedimentos tem por finalidade evitar que os sistemas corporativos do Exército Brasileiro sejam colocados em produção com *bugs* de segurança presentes no seu código. Tudo da maneira mais eficiente e, portanto, menos custosa possível.

3.1 Visão Geral

Tendo em vista que o custo da detecção de falhas de segurança aumenta ao longo do tempo, deve-se planejar e trabalhar com a segurança da informação desde a primeira fase do ciclo de vida do desenvolvimento de *software*, a fim de evitar-se riscos de segurança e custos adicionais de correções de falhas no sistema. Quanto mais tarde no ciclo de vida do sistema uma falha for detectada, mais caro será a sua correção. Portanto, a segurança deve estar intimamente relacionada e envolvida com o ciclo de vida do desenvolvimento de *software*. Ela deve ser implementada e testada já na fase de desenvolvimento e não somente quando o sistema está em produção e os riscos, assim como os custos, são exponencialmente maiores.

A proposta de procedimentos de segurança deste trabalho abrange todas as fases do ciclo de vida e é composta de quatro itens principais: treinamento, revisão do código-fonte, verificação automatizada de vulnerabilidades no código e casos de teste de segurança e vulnerabilidade. Todos esses controles, assim como a sua disposição ao longo do ciclo de vida do desenvolvimento do *software*, podem ser visualizados na figura 3.1.

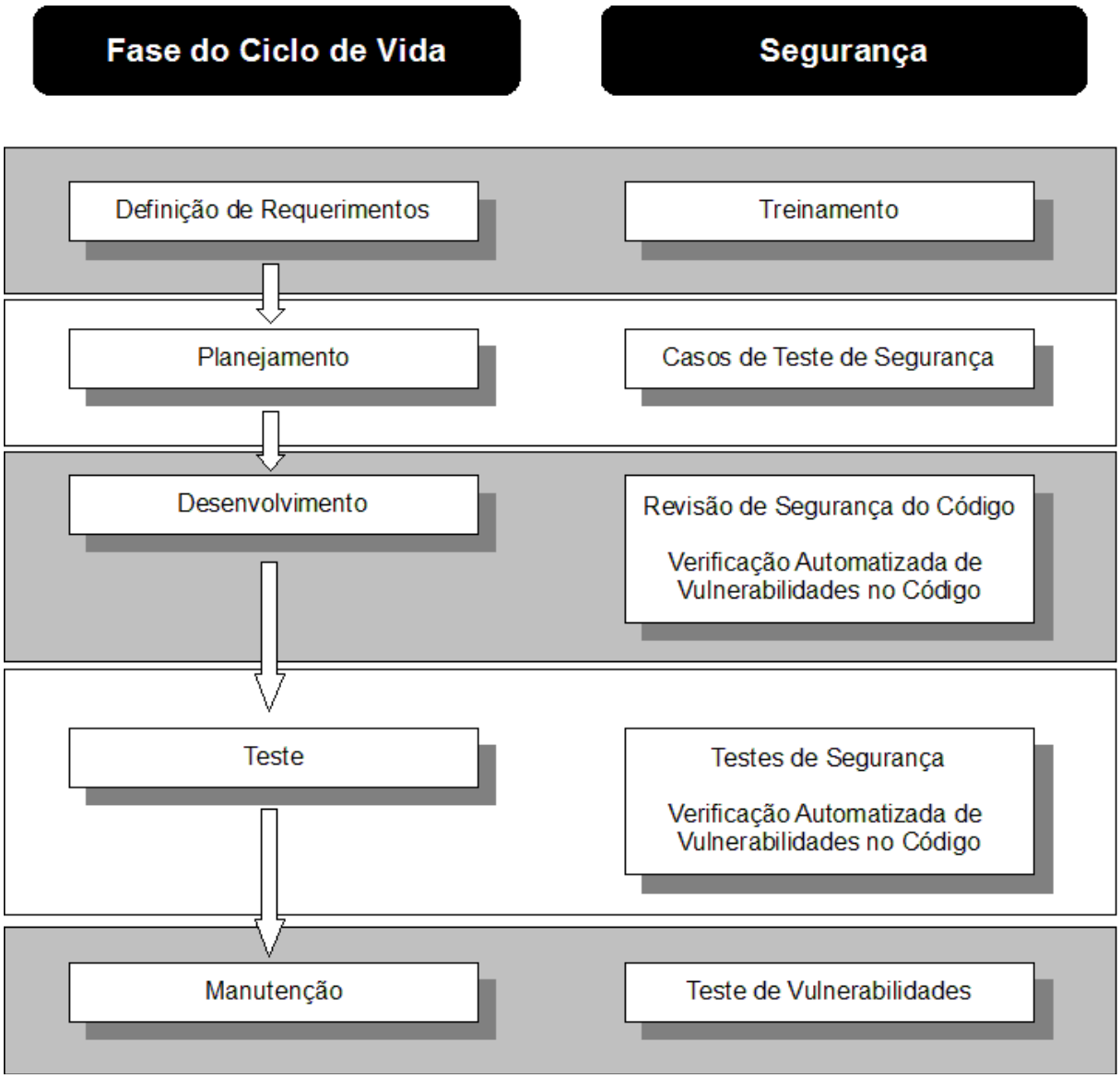

Figura 3.1: Segurança no Ciclo de Vida do Desenvolvimento de *Software*

Fonte: elaborado pelo autor

Analisando a figura, verifica-se que o treinamento deve ser realizado logo no início do projeto, ou seja, durante a fase de definição de requerimentos. Os casos de teste de segurança, por sua vez, devem ser planejados antes mesmo da codificação da primeira linha do *software*, na fase de planejamento. Durante o desenvolvimento são implementados dois controles de segurança: a revisão de segurança do código e a verificação automatizada de vulnerabilidades no código; este último controle é executado novamente durante a fase de teste, desta vez, pela equipe de testadores. A fase de teste também inclui a execução dos testes de segurança, que

haviam sido elaborados durante a fase de planejamento. Finalmente, durante a fase de manutenção do *software*, são executados, periodicamente, testes de vulnerabilidades contra o sistema corporativo.

3.2 Treinamento

O treinamento é o primeiro e mais importante passo para a obtenção de aplicações mais seguras: qualificar o pessoal (desenvolvedores) com treinamento sobre os conceitos básicos de segurança da informação, conforme apresentado neste trabalho (Capítulo 2 - Segurança da Informação). "O nível de segurança de uma corrente é equivalente à resistência oferecida pelo elo mais fraco. O *peopleware* representa justamente esse elo; por isso deve ser alvo de um programa contínuo e dinâmico, capaz de manter os recursos humanos motivados a contribuir, conscientes de suas responsabilidades e preparados para agir diante de antigas e novas situações de risco" (SÊMOLA, 2003, p. 136). Com *peopleware*, o autor citado refere-se aos recursos humanos da empresa. No caso de interesse particular deste trabalho, o *peopleware* é constituído pelos desenvolvedores de *software* do Exército Brasileiro. Ou seja, não adianta ter ferramentas de desenvolvimento modernas e boas políticas de segurança - os elos fortes da corrente - se o pessoal não está treinado para utilizar essas ferramentas de forma adequada, nem qualificado para seguir as políticas adotadas. Por isso, o treinamento do pessoal é indispensável para forjar a resistência do elo constituído pelo *peopleware*.

A importância de treinar o pessoal em segurança da informação também é salientada no parágrafo único do artigo 3º do Decreto número 4.553, de 27 de dezembro de 2002, que diz: "Toda autoridade responsável pelo trato de dados ou informações sigilosos providenciará para que o pessoal sob suas ordens conheça integralmente as medidas de segurança estabelecidas, zelando pelo seu fiel cumprimento". Não menos importante, o caput do referido artigo determina que "A produção, manuseio, consulta, transmissão, manutenção e guarda de dados ou informações sigilosos observarão medidas especiais de segurança". Portanto, um treinamento que qualifique o pessoal responsável pelo trato de informações sigilosas nas medidas de segurança estabelecidas é um requerimento legal.

O conteúdo do presente trabalho pode servir como referência para a elaboração de um

treinamento interno específico de segurança, para o público alvo de programadores do Exército Brasileiro. Os assuntos mais propícios a serem abordados numa instrução sobre segurança da informação são, justamente, os conceitos apresentados na seção Conceitos Fundamentais de Segurança da Informação do capítulo 2 - Segurança da Informação; nominalmente, Confidencialidade, Integridade, Segurança, Autenticação, Autorização, Auditoria, Criptografia, *Hashing* e a classificação das informações conforme o seu grau de sigilo, segundo o Governo Federal. Cada um desses assuntos é abordado em maiores detalhes no apêndice A - Material Didático sobre Desenvolvimento de Aplicações Seguras, que foi escrito com a intenção de servir como base para as notas de aula do treinamento de segurança.

Propõe-se que o treinamento seja intitulado de "Instrução de Desenvolvimento de Aplicações Seguras", e que tenha como público alvo todos os programadores de *software* do EB. O treinamento deve ser presencial, composto de uma parte teórica dos conceitos fundamentais de segurança anteriormente mencionados; mais um laboratório prático, com a aplicação prática dos conceitos teóricos. A duração sugerida para o treinamento é de duas semanas. Os quadros a seguir expõe o conteúdo do treinamento; o quadro 3.1 detalha o conteúdo da parte teórica, enquanto o quadro 3.2 lista os tópicos do laboratório prático.

UNIDADE DIDÁTICA I - CONCEITOS FUNDAMENTAIS DE SEGURANÇA DA INFORMAÇÃO (continua)		
ASSUNTOS	**OBJETIVOS ESPECÍFICOS**	**QTDE HORAS**
1. CIA	a. Assimilar a importância do conceito. b. Definir os conceitos de confidencialidade, integridade e disponibilidade. c. Saber exemplificar situações em que cada conceito é empregado. d. Identificar quais técnicas de computação são utilizadas em cada conceito.	6
2. AAA	a. Assimilar a importância do conceito. b. Definir os conceitos de autenticação, autorização e auditoria. c. Saber exemplificar situações em que cada conceito é empregado. d. Identificar quais técnicas de computação são utilizadas em cada conceito.	6
3. Criptografia	a. Definir o conceito de criptografia.	4

UNIDADE DIDÁTICA I - CONCEITOS FUNDAMENTAIS DE SEGURANÇA DA INFORMAÇÃO (conclusão)		
ASSUNTOS	**OBJETIVOS ESPECÍFICOS**	**QTDE HORAS**
	b. Conhecer as técnicas de criptografia existentes. c. Explicar e diferenciar a criptografia simétrica da assimétrica, citando as vantagens e desvantagens de cada uma. d. Explicar como a criptografia é utilizada para garantir a confidencialidade, integridade, autenticidade e não-repudiação.	
4. *Hashing*	a. Explicar o que é uma função *hash*. b. Entender a diferença entre *hashing* e criptografia c. Exemplificar situações em que o *hash* e a criptografia. devem ser utilizados. d. Citar as vantagens e desvantagens e, em função dessas, explicar por que cada método é utilizado.	2
5. Classificação Grau de Sigilo	a. Conhecer os graus de classificação ultra-secreto, secreto, confidencial, reservado. b. Exemplificar dados ou informações para cada tipo de classificação de sigilo.	2
INSTRUÇÕES METODOLÓGICAS		
a. Palestra teórica sobre os conceitos fundamentais. b. Exemplos práticos de aplicação dos conceitos. c. Notícias de eventos relacionados ocorridos recentemente.		

Quadro 3.1: Conteúdo teórico da instrução sobre desenvolvimento de aplicações seguras.

UNIDADE DIDÁTICA II - LABORATÓRIO DE SEGURANÇA DA INFORMAÇÃO (continua)		
ASSUNTOS	**OBJETIVOS ESPECÍFICOS**	**QTDE HORAS**
1. *SQL Injection*	a. Assimilar conhecimentos necessários de SQL. b. Compreender o funcionamento do banco de dados. c. Saber utilizar corretamente *views* e *stored procedures*. d. Entender o funcionamento do ataque. e. Identificar qual a vulnerabilidade explorada pelo atacante. f. Realizar o ataque em ambiente de laboratório. g. Conhecer os métodos de defesa contra o ataque.	4
2. *Cross-site Scripting*	a. Conhecer o funcionamento de *scripts* para *websites*. b. Entender o funcionamento do ataque.	4

UNIDADE DIDÁTICA II - LABORATÓRIO DE SEGURANÇA DA INFORMAÇÃO (conclusão)		
ASSUNTOS	**OBJETIVOS ESPECÍFICOS**	**QTDE HORAS**
	c. Identificar qual a vulnerabilidade explorada pelo atacante. d. Realizar o ataque em ambiente de laboratório. e. Conhecer os métodos de defesa contra o ataque.	
3. *Buffer Overflow*	a. Adquirir os conhecimentos necessários de linguagens de programação. b. Explicar o funcionamento de um *buffer*. c. Citar em quais situações um *buffer* é utilizado. d. Entender o funcionamento do ataque. e. Identificar qual a vulnerabilidade explorada pelo atacante. f. Realizar o ataque em ambiente de laboratório. g. Conhecer os métodos de defesa contra o ataque.	4
4. *Fishing & Farming*	a. Explicar o que é *fishing* e *farming*. b. Saber as técnicas de engenharia social. c. Entender o funcionamento do ataque. d. Identificar qual a vulnerabilidade explorada pelo atacante. e. Realizar o ataque em ambiente de laboratório. f. Conhecer os métodos de defesa contra o ataque.	4
5. Formas de Proteção	a. Entender os mecanismos genéricos de proteção contra os ataques	4
INSTRUÇÕES METODOLÓGICAS		
a. Laboratório prático. b. Instrutor realiza exemplo passo-a-passo de um ataque. c. Discente executa também todos os passos em seu computador. d. Dispor um discente por computador. e. Trabalho individual: cada discente deve realizar os exercícios e não apenas assistir.		

Quadro 3.2: conteúdo prático da instrução sobre desenvolvimento de aplicações seguras.

3.3 Revisão do Código-Fonte

Com o pessoal qualificado e, principalmente, conscientizado, pelo treinamento de desenvolvimento de aplicações seguras, essa etapa das políticas de segurança constitui-se em adotar as técnicas aprendidas durante o treinamento no dia-a-dia do trabalho de

desenvolvimento de *software*. Para facilitar tal tarefa, é apresentado a seguir uma série de itens a serem verificados pelo programador durante o desenvolvimento do *software.* Os itens são apresentados em quadros, organizados conforme os conceitos fundamentais de segurança vistos no capítulo 2 - Segurança da Informação. O quadro 3.3 contém os controles de confidencialidade. O quadro 3.4, por sua vez, apresenta os controles de integridade. Já o quadro 3.5 lista os controles de disponibilidade. E, finalmente, o quadro 3.6 mostra os controles de autenticação, autorização e auditoria. Sugere-se que esses itens sejam acrescentados ao teste de *peer-review* adotado nas seções de informática das OM.

CONFIDENCIALIDADE	
Processamento	- mostrar apenas as informações necessárias; - mascarar os dados para exibir.
Trânsito	- criptografar dados confidenciais; - utilizar tunelamento seguro (protocolo *https*).
Armazenamento	- dados ultra-secretos devem ser armazenados criptografados; - senhas devem ser transformadas em códigos *hash* antes de serem armazenadas; - nunca armazenar senhas em texto aberto.

Quadro 3.3: controles de confidencialidade.

INTEGRIDADE	
Validar todas as entradas de dados.	- garantir que os dados são recebidos pelo sistema de forma correta e validada; - verificar o tamanho máximo dos campos de entrada.
Falhar de forma segura.	- não permitir autenticação automática após tentativa falhada; - não manter credenciais em *cache;* - registrar em *log* a ocorrência da falha para auditoria posterior.
Manipulação de Erros	- exibir mensagem de erro genérica;

Quadro 3.4: controles de integridade.

DISPONIBILIDADE
- manter duplicação de dados em locais de armazenamentos diferentes; - realizar cópias de segurança (*backups*) regularmente; - procedimento escrito para a restauração dos dados em caso de falha.

Quadro 3.5: controles de disponibilidade.

A^3	
AUTENTICAÇÃO	- armazenar senhas na forma de *hash*; - nunca guardar informação de autenticação no cliente; - não armazenar senhas em *cookies;* - não utilizar o autenticador de seção (*session id*) como único; mecanismo de autenticação de usuário (previne ataques *hijack* e *replay*); - negar acesso anônimo.
AUTORIZAÇÃO	- utilizar como regra geral: permissão negada; - segregar, no mínimo, o grupo de administradores do sistema; - contas de usuário com as permissões mínimas para realizar a função; - contas de processo e serviço não devem ter privilégios elevados; - aplicações e processos não devem utilizar contas administrativas padronizadas (*admin, sa, dba*); - acesso das aplicações ao banco de dados não deve ser direto; - acessar o banco de dados através de *stored procedures* ou *views;* - o acesso a informações de configuração da aplicação deve ser restrito; - nenhum *backdoor* ou acesso privilegiado não autenticado deve ser desenvolvido, habilitado ou utilizado.
AUDITORIA	- acessos e ações de administrador devem ser registrados em *log;* - armazenar em registro transações críticas; - anexar, sempre; sobrescrever ou apagar, nunca.

Quadro 3.6: controles de autenticação, autorização e auditoria.

Adicionalmente, salienta-se que o sub-sistema de *logs* é indispensável, pois é de extrema importância para detectar a ocorrência de uma falha de segurança, possibilitar a análise e compreensão do ataque e, por fim, o saneamento da vulnerabilidade. Também deve-se evitar o uso de rotinas conhecidamente inseguras, especialmente as de manipulação de *strings* nas linguagens C e C++. Com relação ao banco de dados, deve-se sempre utilizar *Stored Procedures* (SP) e *views*. Da mesma forma, na montagem de *strings* de comandos de SQL, deve-se tomar as devidas precauções de validação de entradas para não sofrer *SQL-Injection* ao formar comandos concatenando *input* do usuário. Ressalta-se, mais uma vez, que dados sensíveis devem ser sempre criptografados durante o trânsito e armazenagem. É necessário, ainda, verificar a adequação do algoritmo de criptografia utilizado para garantir o nível de proteção adequado à classificação de sigilo da informação em questão. Por fim, implementar e executar periodicamente mecanismos de *backup,* ou seja, realizar regularmente de cópias de segurança das informações armazenadas pelo sistema.

3.4 Verificação Automatizada

Essa verificação constitui-se em utilizar uma ferramenta automatizada de verificação de falhas de segurança no código fonte dos sistemas corporativos. A verificação manual de uma série de itens contra todas as linhas de código de um programa, se realizada por um operador humano, tornar-se-ia uma tarefa repetitiva, tediosa, cansativa e, assim, propensa a erros; além disso, consumiria muito tempo para ser concluída com sucesso. Tendo isso em vista, verifica-se que o uso de uma ferramenta automatizada para a realização de tal tarefa altamente adequada e aconselhável.

Atualmente existem diversas opções desse tipo de ferramenta disponíveis, tanto comerciais como de *software* livre. Das opções comerciais pagas, a ferramenta *Fortify* (FORTIFY, 2009) constitui, até a data de elaboração do presente trabalho, a melhor escolha. Essa ferramenta é composta por três verificadores que detectam as vulnerabilidades do programa. O *Fortify* inclui um módulo para ajudar os desenvolvedores a sanar as vulnerabilidades identificadas. Adicionalmente, a ferramenta também possui um console de relatórios e gerenciamento.

Entre as opções de *software* livre, a ferramenta YASCA é uma escolha popular entre os programadores (YASCA, 2009). O nome da ferramenta é a sigla, em inglês, de *Yet Another Source Code Analyzer*, ou seja, "mais um analisador de código fonte". YASCA é um programa de código aberto que procura por vulnerabilidades de segurança, verifica a qualidade do código, o seu desempenho, e a conformidade do mesmo com as melhores práticas estabelecidas de desenvolvimento de código. Ao contrário da opção comercial paga, a ferramenta não possui uma interface muito elaborada; seu funcionamento é via linha-de-comando e a apresentação dos seus resultados é através de um arquivo HTML.

Essa etapa de verificação do código fonte tem o potencial de causar uma melhora significativa na qualidade do *software* produzido em termos de segurança. Mas requer a colaboração dos programadores para ser colocada em prática, pois são os desenvolvedores de cada sistema que devem executar a verificação no código produzido e realizar a correção das falhas encontrada. Adicionalmente, tanto as opções comerciais como as de código livre apresentam um certo índice de falsos-positivos em seus resultados. Um falso-positivo é um trecho de código que a ferramenta julga ser uma falha de segurança mas que, na realidade,

não é um *bug* de segurança. Portanto, é necessário uma revisão manual dos resultados apresentados pela ferramenta para a verificação dos falsos-positivos.

Nota-se que existe uma dificuldade em potencial na adoção dessa etapa dos procedimentos de segurança. A prática demonstra que, apesar de eficientes, esse tipo de ferramenta encontra certa resistência de ser utilizada pelos programadores, pois: acarreta mais trabalho para os mesmos com a correção dos *bugs* encontrados; e, principalmente, o temor (na maioria das vezes, não fundamentado) dos desenvolvedores de que a detecção e exposição de falhas de segurança no seu código coloque em dúvida a qualidade do seu trabalho. A melhor forma de enfrentar essa dificuldade é com treinamento, pois a qualificação vai conscientizar o programador da importância do cuidado com segurança da informação, bem como demonstrar a utilidade de ferramenta que, na realidade, vai auxiliar o desenvolvedor a melhorar a qualidade do seu código. Assim, ao invés de temer a ferramenta, o programador deve utilizá-la como atestado da qualidade do código que ele produz.

3.5 Testes de Segurança e Vulnerabilidade

Existem dois tipos diferentes de testes a serem executados: os testes de segurança e os testes de vulnerabilidades. Esses testes tem por objetivo verificar se nenhuma falha de segurança passou pelos controles anteriores de segurança da informação (TIPTON, KRAUSE, 2001). A eficiência e confiabilidade desses testes baseia-se no princípio de que quem programa um sistema não testa o mesmo; ou seja, os testes devem ser conduzidos por pessoal específico - os testadores - e não pelos programadores. Salienta-se, portanto, a importância de que pessoas diferentes testem o sistema que foi implementado.

Os Testes de Segurança são elaborados na fase de planejamento do projeto, ou seja, antes mesmo do código-fonte ser implementado. Diferem do teste normal pois não estão focados nas funcionalidades de negócio do programa mas, sim, em potenciais problemas de segurança. A sua execução será realizada concomitantemente com os teste funcionais, somente após a finalização da codificação e o início da fase de teste propriamente dita. Esses testes são fundamentados na primeira parte do treinamento de desenvolvimento de aplicações seguras, ou seja, nos conceitos fundamentais de segurança. São testes de fundamentação

teórica, e verificam os controles de confidencialidade, integridade e disponibilidade que devem estar presentes no código do programa, assim como os controles de autenticação, autenticidade e auditoria. Os testes de segurança são elaborados a partir de casos de teste baseados nos requerimentos do projeto em particular.

O teste de vulnerabilidade, por sua vez, é executado periodicamente após a implantação do sistema; ou seja, esse teste é executado em ambiente de produção. Seu propósito é justamente testar o sistema contra vulnerabilidades quando o mesmo está inserido dentro de seu ambiente de operação, levando-se em consideração o sistema operacional, a rede e o próprio sistema em si. Devido a sua natureza periódica, esse tipo de teste tem por função capturar defeitos injetados no sistema a partir de atualizações de código do programa, ou mesmo a partir de mudanças realizadas no ambiente em que o sistema executa, como atualizações do sistema operacional ou mudanças no banco de dados. Os testes de vulnerabilidades são elaborados com base na segunda parte da instrução de desenvolvimento de aplicações seguras, isto é, a parte prática de laboratório. Basicamente, os testes de vulnerabilidades são a execução, na prática, dos exercícios realizados em laboratório durante o treinamento. Salienta-se, entretanto, a importância do testador em manter-se sempre atualizado com relação a novas técnicas de ataque e verificação de vulnerabilidade, assim como a importância do treinamento ser constantemente atualizado, para que passe a cobrir as novas técnicas à medida que as mesmas forem surgindo.

Um exemplo de teste de segurança, para um sistema que trabalhe com informações classificadas como secretas, seria verificar se os controles confidencialidade estão implementados, ou seja, testar se o programa está criptografando adequadamente as informações secretas que armazena. Nesse exemplo em particular, seria necessário verificar se o algoritmo de criptografia utilizado é considerado seguro, ou seja, se ele não foi quebrado ou se não possui vulnerabilidades conhecidas em sua implementação. Já um teste de vulnerabilidade seria, por exemplo, verificar se o *software* está vulnerável a um ataque de *SQL-Injection*, seja esse decorrente da programação original do sistema, ou advinda de alguma atualização do código feita posteriormente, ou mesmo decorrente de mudanças nas *stored procedures* no banco de dados do sistema. Por isso, é importante realizar periodicamente o teste de vulnerabilidades: para que sejam detectadas possíveis falhas introduzidas em atualizações funcionais do sistema.

3.6 Considerações Adicionais

Salienta-se a necessidade de conscientização do comando com relação à importância de uma política, ou regulamento, para a devida adoção dos cuidados de segurança indispensáveis durante o ciclo de vida do desenvolvimento de *software*. "Somente com apoio executivo as ações de segurança ganharão autonomia e abrangência capazes de incidir corporativamente sobre os frutos de segurança" (SÊMOLA, 2003, p. 40).

Em artigo publicado na revista de produção científica da ESAEX, Santos, Silva e Nalin (2006) propõe a criação de um grupo de segurança da informação em cada organização militar. Tal medida é válida e, de fato, possui o potencial para melhorar a segurança da informação no âmbito de cada OM. Mas também existe a necessidade organizar e padronizar os procedimentos de segurança da informação no Exército como um todo. Portanto, um órgão geral ou grupo de trabalho para a coordenação dessas atividades específicas no EB é uma boa prática de segurança.

4 VALIDAÇÃO

No capítulo 3 - Procedimentos de Segurança, foi apresentada a proposta deste trabalho com relação aos procedimentos de segurança de informação a serem adotados para os sistemas corporativos do Exército Brasileiro. Foi visto que é necessário treinar o pessoal, bem como adotar novos procedimentos e ferramentas que auxiliam na qualidade do desenvolvimento de *software*. A implementação de todos esses controles, assim como a sua validação, exige tempo - no mínimo, um ciclo de vida completo de desenvolvimento de *software*. Infelizmente, o período disponível para a conclusão do presente trabalho não comporta todo o tempo necessário para tal atividade. Não obstante, este capítulo demonstra como validar os procedimentos de segurança propostos, dado o tempo e os recursos necessários.

Inicialmente, é apresentada uma visão geral da validação dos procedimentos. Em seguida, a validação de cada procedimento é planejada, com o detalhamento de cada fase. Inclui-se no planejamento apresentado a seguir, a discriminação do tempo necessário para a execução de cada uma das etapas do procedimento de validação. O questionário necessário para a validação encontra-se no apêndice B - Questionário de Avaliação de Conhecimentos em Segurança da Informação.

4.1 Procedimentos de Validação

A validação das propostas de segurança da informação apresentadas neste trabalho exige a aplicação dos procedimentos, a mensuração da eficácia dos mesmos e a análise dos resultados obtidos. Assim, a validação é dividida em duas grandes etapas: a primeira é constituída pela aplicação da instrução de desenvolvimento de aplicações seguras e a mensuração do aproveitamento do pessoal com relação ao treinamento; a segunda etapa, por sua vez, é composta pela aplicação na prática dos procedimentos ministrados na instrução,

bem como pela mensuração da eficácia dos procedimentos em comparação com os métodos anteriores. O quadro 4.1, a seguir, detalha cada uma das atividades de cada etapa da validação.

Etapa	Atividade	Tempo Necessário
1	Questionário de Avaliação de Conhecimentos	1 hora
	Instrução de Desenvolvimento de Aplicações Seguras	2 semanas
	Questionário de Avaliação de Conhecimentos (2ª aplicação)	1 hora
2	Medir aplicações previamente desenvolvidas	1 Fase de Teste
	Desenvolvimento de Aplicações Seguras	1 Ciclo de Vida de Desenvolvimento de *Software*
	Revisão do Código-Fonte	
	Análise Automatizada do Código-Fonte	
	Testes de Segurança e Vulnerabilidade	
	Mensurar aplicação desenvolvida com as propostas deste trabalho	

Quadro 4.1: cronograma de atividades para a validação das propostas.

4.2 Primeira Etapa

A primeira etapa da validação constitui-se, basicamente, na realização da instrução de desenvolvimento de aplicações seguras, com a aplicação de questionários de avaliação de conhecimentos para verificar-se a eficácia da instrução ministrada. A validação deve ocorrer conforme descrito a seguir.

Inicialmente, selecionar um grupo de desenvolvedores de *software* do Exército Brasileiro. Aplicar, antes da realização do treinamento, o questionário de avaliação de conhecimentos em Segurança da Informação (vide apêndice B - Questionário de Avaliação de Conhecimentos em Segurança da Informação). Em seguida, realizar o treinamento de Desenvolvimento de Aplicações Seguras com base no material didático apresentado no apêndice A - Material Didático sobre Desenvolvimento de Aplicações Seguras - do presente trabalho. Ao término do curso, aplicar novamente o mesmo questionário. A partir da comparação dos resultados obtidos no questionário antes e depois da ministração da instrução, será possível medir a melhora do nível de conhecimento técnico do pessoal com relação ao

assunto. Com base nas tendências apresentadas no questionário, também será possível efetuar ajustes e melhorias específicas na instrução.

4.3 Segunda Etapa

A segunda etapa inicia-se com a mensuração das aplicações previamente desenvolvidas pelo grupo de desenvolvedores de *software* selecionados durante a primeira etapa. Para cada uma das aplicações a serem analisadas, será necessário investir o tempo correspondente a uma fase de teste do ciclo de vida da aplicação em particular. A mensuração em si consistem em executar a ferramenta automatizada de verificação de vulnerabilidades no código e analisar os resultados apresentados, filtrando os falsos-positivos, conforme explanado na seção 3.4 - Verificação Automatizada - do capítulo 3 - Procedimentos de Segurança.

A seguir, é desenvolvida uma nova aplicação. Desta vez, com o pessoal devidamente treinado e seguindo as propostas apresentadas neste trabalho; ou seja, os programadores realizarão a revisão do código-fonte e, também, executarão a ferramenta automatizada para detecção de vulnerabilidades na aplicação. Da mesma forma, também serão devidamente planejados e executados os testes de segurança e de vulnerabilidade do sistema. Todos esses procedimentos consumirão um ciclo de vida de desenvolvimento de *software* completo para serem implementados. Ao término do ciclo, a mesma mensuração executada no início da etapa, para aplicações anteriormente desenvolvidas, deverá ser realizada para a nova aplicação desenvolvida com as propostas de segurança deste trabalho. Com base nesses dois índices, será possível comparar e medir a eficiência dos controles de segurança. Validar-se-ia, assim, a proposta do presente trabalho.

5 CONCLUSÃO

Atualmente, a maioria dos ataques a sistemas informatizados ocorrem através de vulnerabilidades nas aplicações corporativas e não mais por falhas de segurança nas redes de computadores. O Exército Brasileiro, que possui diversos sistemas corporativos que o auxiliam a cumprir sua missão constitucional, precisa adotar medidas de segurança da informação para sua proteção. Assim, torna-se necessário direcionar o foco das medidas de segurança para o desenvolvimento de *software*. Adicionalmente, sabe-se que corrigir defeitos de *software* é muito mais caro depois que o sistema já está em produção do que quando ele está em desenvolvimento. Assim, é mais fácil e menos custoso corrigir as falhas de segurança durante a fase de codificação do aplicativo. Por isso, os procedimentos de segurança propostos neste trabalho foram voltados para o desenvolvimento de aplicações seguras, incluindo controles em todas as fases do ciclo de vida do desenvolvimento do *software*.

Na proposta apresentada neste trabalho, os procedimentos de segurança da informação incluem o treinamento de pessoal, através de instrução especialmente elaborada para o Exército Brasileiro; a revisão do código-fonte, com o auxílio de uma série de controles de segurança listados e explanados neste trabalho; a verificação automatizada de vulnerabilidades no código, por meio de ferramentas específicas sugeridas neste texto; e, por fim, casos de teste de segurança e vulnerabilidade, com a elaboração e aplicação dos testes nas fases apropriadas do projeto.

A validação da proposta de procedimentos de segurança da informação para os sistemas corporativos do Exército Brasileiro exige o treinamento de um grupo de programadores através da instrução de desenvolvimento de aplicações seguras e a medição, através de questionário, da eficácia do treinamento; também é necessário auferir o nível de segurança das aplicações previamente desenvolvidas pelo grupo de desenvolvedores, a fim de medir a melhora obtida com a adoção dos procedimentos propostos. Dessa forma, comprovar-se-á que a execução das propostas de segurança pode garantir propriedades desejáveis, como confidencialidade e integridade dos dados, eliminar vulnerabilidades do sistema corporativo e reduzir riscos de segurança do *software*.

A contribuição deste trabalho, para o Exército Brasileiro, está no fortalecimento de seus sistemas corporativos, com a adoção das medidas propostas. Também contribui para a qualificação do seu pessoal, com a implantação da instrução sugerida. Com isso, abre-se a possibilidade de se criar uma cultura de segurança da informação no desenvolvimento de sistemas para o EB. No âmbito geral da segurança da informação, este trabalho apresenta a sua parcela de contribuição em uma área que só recentemente vem sendo trabalhada pela comunidade especializada, isto é, a exploração e defesa de vulnerabilidades nos sistemas corporativos e não apenas das redes de computadores.

Finalmente, devido ao tempo exigido para ministrar a instrução de treinamento proposta, acrescida do período exigido para a realização do desenvolvimento de *software*, não foi possível executar a validação planejada dentro da janela de tempo disponível; para tanto, seria necessário empregar um ciclo de vida de desenvolvimento de *software* completo. Tendo em vista essa limitação, sugere-se como trabalho futuro a aplicação da instrução de Desenvolvimento de Aplicações Seguras, com base no material e cronograma apresentados neste texto. Com a realização do treinamento, poder-se-á desenvolver outro trabalho futuro: a validação das propostas de segurança, conforme descrito neste trabalho.