MINISTÉRIO DA DEFESA
COMANDO DA AERONÁUTICA

TECNOLOGIA DA INFORMAÇÃO

ICA 7-32

REQUISITOS TÉCNICOS PARA AQUISIÇÃO E DESENVOLVIMENTO DE SISTEMAS SEGUROS DE TECNOLOGIA DA INFORMAÇÃO DO DECEA

2014

MINISTÉRIO DA DEFESA
COMANDO DA AERONÁUTICA
DEPARTAMENTO DE CONTROLE DO ESPAÇO AÉREO

TECNOLOGIA DA INFORMAÇÃO

ICA 7-32

REQUISITOS TÉCNICOS PARA AQUISIÇÃO E DESENVOLVIMENTO DE SISTEMAS SEGUROS DE TECNOLOGIA DA INFORMAÇÃO DO DECEA

2014

MINISTÉRIO DA DEFESA
COMANDO DA AERONÁUTICA
DEPARTAMENTO DE CONTROLE DO ESPAÇO AÉREO

PORTARIA DECEA Nº 76/DGCEA, DE 24 DE JUNHO DE 2014.

Aprova a edição da Instrução de Requisitos Técnicos para Aquisição e Desenvolvimento de Sistemas Seguros de Tecnologia da Informação do DECEA.

O DIRETOR-GERAL DO DEPARTAMENTO DE CONTROLE DO ESPAÇO AÉREO, no uso das atribuições que lhe conferem o inciso IV do art. 195, do Regimento Interno do Comando da Aeronáutica, aprovado pela Portaria nº 1.049/GC3, de 11 de novembro de 2009, e o inciso IV do art. 10 do Regulamento do DECEA, aprovado pela Portaria nº 1.668/GC3, de 16 de setembro de 2013, resolve:

Art.1º Aprovar a edição da ICA 7-32 "Requisitos Técnicos para Aquisição e Desenvolvimento de Sistemas Seguros de Tecnologia da Informação do DECEA", que com esta baixa.

Art. 2º Esta Instrução entra em vigor na data de sua publicação.

Ten Brig Ar RAFAEL RODRIGUES FILHO
Diretor-Geral do DECEA

(Publicado no BCA nº 122, de 2 de julho de 2014)

SUMÁRIO

1 DISPOSIÇÕES PRELIMINARES

1.1 FINALIDADE

Esta Instrução tem por finalidade estabelecer recomendações técnicas para a aquisição e o desenvolvimento de sistemas de Tecnologia da Informação de forma a atender parâmetros desejáveis de autenticidade, confidencialidade, integridade e disponibilidade relativos aos sistemas de informação do Departamento de Controle do Espaço Aéreo e Organizações Militares subordinadas.

1.2 ÂMBITO E GRAU DE SIGILO

Esta Instrução se aplica ao DECEA e a todas as suas Organizações Militares Subordinadas, sendo considerado ostensivo o seu grau de sigilo.

1.3 SIGLAS E ABREVIATURAS

API - Interface de Programação de Aplicativos
COTS - *Commercial Off The Shelf* – pronto para uso
CSRF - *Cross Site Request Forgery*
DECEA - Departamento de Controle do Espaço Aéreo
FSR - Revisão Final de Segurança
HTTP - Protocolo de transferência de hipertexto
HTTPS - Protocolo de transferência de hipertexto seguro
ICA - Instrução de Comando da Aeronáutica
IDS - *Intrusion Detection System* – Sistema de Detecção de Intrusão
IEC - *International Electrotechnical Commission*
IPS - *Intrusion Prevention System* – Sistema de Prevenção de Intrusão
ISO - *International Organization for Standardization*
OM - Organização Militar
SDL - Ciclo de Vida do Desenvolvimento da Segurança
SDTE - Subdepartamento Técnico
SI - Segurança da Informação
SISCEAB - Sistema de Controle do Espaço Aéreo Brasileiro
SQL - Linguagem de Consulta Estruturada
SSL - Protocolo de Camada de *Sockets* Segura
TI - Tecnologia da Informação
TLS - Segurança da Camada de Transporte
URL - Localizador-Padrão de Recursos

1.4 CONCEITUAÇÕES

1.4.1 BIBLIOTECA DE PROGRAMAS-FONTE

É um conjunto de todos os arquivos que são necessários para se compilar, construir e documentar uma aplicação.

1.4.2 *BUFFER OVERRUN/OVERFLOW*

É uma anomalia no qual um programa, ao escrever dados em um *buffer*, ultrapassa seus limites e sobrescreve na área de dados da memória adjacente.

1.4.3 *BUG BAR*

É usada para definir os limites de severidade das vulnerabilidades de segurança, por exemplo: nenhuma vulnerabilidade conhecida na aplicação com uma classificação "crítica" ou "importante" no momento da liberação.

1.4.4 *COMMON CRITERIA*

Norma internacional (ISO/IEC 15.408, 2009) que fornece uma base de critérios comuns para a avaliação das propriedades de segurança de produtos e sistemas de TI.

1.4.5 *COVERT CHANNELS*

Os *covert channels* são caminhos não previstos que podem conduzir um fluxo de informações maliciosas, de forma oculta. A exploração destes canais frequentemente é realizada por cavalos de Troia.

1.4.6 CSRF- *CROSS-SITE REQUEST FORGERY*

Um ataque CSRF força a vítima que possui uma sessão ativa em um navegador a enviar uma requisição HTTP forjada, incluindo o *cookie* da sessão da vítima e qualquer outra informação de autenticação incluída na sessão, a uma aplicação Web vulnerável. Esta falha permite ao atacante forçar o navegador da vítima a criar requisições que a aplicação vulnerável aceite como requisições legítimas realizadas pela vítima (OWASP, 2013).

1.4.7 DESENVOLVIMENTO SEGURO

Refere-se a todas as fases do ciclo de vida do desenvolvimento de um sistema seguro.

1.4.8 *HASH*

Refere-se a uma função que transforma uma sequência de caracteres em uma nova sequência de caracteres conhecida como síntese da mensagem.

1.4.9 HTTP GET E POST

O protocolo de transferência de hipertexto (HTTP) é projetado para permitir a comunicação entre clientes e servidores. Dois métodos comumente usados para uma solicitação entre um cliente e servidor são: GET e POST. GET é menos seguro comparado ao POST porque dados são enviados como parte da URL.

1.4.10 HTTPS

O HTTPS (*HyperText Transfer Protocol Secure* - protocolo de transferência de hipertexto seguro) é uma implementação do protocolo HTTP sobre uma camada adicional de segurança que utiliza o protocolo SSL/TLS. O protocolo HTTPS é utilizado, em regra, quando se deseja evitar que a informação transmitida entre o cliente e o servidor seja visualizada por terceiros.

1.4.11 IDS/IPS

Um IDS/IPS é um equipamento ou *software* que inspeciona o tráfego e procura assinaturas de ataques ou padrões de comportamento incomum. Um IDS alerta o administrador do sistema por *pager*, *e-mail*, ou telefone celular quando um evento que aparece na lista de eventos de segurança da empresa é acionado. Sistemas de prevenção de intrusão (IPS) iniciam contramedidas, tais como o bloqueio do tráfego quando um tráfego suspeito é detectado.

1.4.12 OWASP

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

Os estudos e documentos da OWASP são disponibilizados para toda a comunidade internacional, e adotados como referência por entidades como *U.S. Defense Information Systems Agency* (DISA), *U.S. Federal Trade Commission*, várias empresas e organizações mundiais das áreas de Tecnologia, Auditoria e Segurança, e também pelo *PCI Council*.

1.4.13 OWASP *TOP* 10

O trabalho mais conhecido da OWASP é sua lista "The Top 10 Most Critical Web Application Security Risks" ou "OWASP TOP 10", que reúne os riscos de ataque mais críticos exploráveis a partir de vulnerabilidades nas aplicações *Web*. Atualizadas a cada três anos, estas listas ficam disponíveis no seguinte *site*: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.

1.4.14 *PATCH*

Um *patch* é uma atualização ou correção de um *software*.

1.4.15 *PREPARED STATEMENT*

Um *prepared statement* é um modelo de instrução, usado por exemplo em consultas ou atualizações em SQL, no qual certos valores constantes são substituídos durante cada execução.

1.4.16 *ROLLBACK*

Um *rollback* ou procedimento de *rollback* é uma estratégia de retorno às condições anteriores que deve ser disponibilizada antes que mudanças sejam implementadas no sistema.

1.4.17 *SALT*

É um dado aleatório que é usado como uma entrada adicional para a função unidirecional que cria o *hash* de uma senha.

1.4.18 SISTEMA DE INFORMAÇÃO

É a expressão utilizada para descrever um sistema, automatizado ou manual, que abrange pessoas, máquinas, ou métodos organizados para coletar, processar, transmitir, disseminar ou armazenar dados que representam informação para o usuário ou cliente.

1.4.19 SISTEMA DE TI

É a expressão utilizada para descrever um sistema de informação automatizado.

1.4.20 SISTEMA SEGURO

É um sistema que possui mecanismos de proteção aos quesitos de confidencialidade, integridade e disponibilidade das informações do sistema e a integridade e a disponibilidade dos recursos de processamento, sob controle do proprietário ou administrador do sistema.

1.4.21 SISTEMAS CRÍTICOS

Sistema de informação em que a falha pode causar graves consequências humanas, econômicas ou de imagem para o DECEA.

1.4.22 SSL/TLS

Ambos são protocolos criptográficos que conferem segurança de comunicação na Internet para serviços como *e-mail* (SMTP), navegação por páginas (HTTPS) e outros tipos de transferência de dados.

1.4.23 SQL

É a linguagem declarativa padrão para bancos de dados relacionais.

1.4.24 USO AMIGÁVEL

Característica que faz o uso de um *software* ser de fácil compreensão para usuários finais.

1.4.25 USUÁRIO FINAL

Usuário, presumidamente sem conhecimento técnico, que opera o sistema diretamente através de sua interface.

1.4.26 VALIDAÇÃO POSITIVA

A validação positiva é um mecanismo que usa critérios pré-definidos para validar o tamanho, caracteres, formato, e as regras de negócio que se aplicam sobre os dados antes de aceitar a entrada. Qualquer dado que não atenda aos critérios deve ser rejeitado.

1.4.27 VULNERABILIDADE DE SEGURANÇA

É uma fragilidade de um ativo ou grupo de ativos que pode ser explorada por uma ou mais ameaças (ABNT NBR ISO/IEC 27002, 2005, item 2.17).

2 CONSIDERAÇÕES INICIAIS

2.1 INTRODUÇÃO

2.1.1 Um produto que não seja adequadamente seguro pode causar sérios prejuízos às informações corporativas por ele mantidas. Portanto, conclui-se que a melhora em seus requisitos de segurança implica um aumento da aceitação e satisfação do cliente, representando um impacto positivo em sua qualidade. Apesar do fato de que a necessidade e a força da segurança são orientadas por contexto – e que situações diferentes demandam soluções diferentes –, o que é claro nesse argumento é que a segurança é um subconjunto da qualidade.

2.1.2 A segurança está relacionada à confiabilidade, mas não é um subconjunto dela. Por exemplo, uma solução que protege dados secretos não precisa necessariamente ser confiável. Se o sistema falhar, mas fizer isso de uma forma que não revele os dados, ele ainda pode ser considerado seguro.

2.1.3 A figura 1 abaixo sintetiza os conceitos apresentados, mostrando que a preocupação com a qualidade e com a confiabilidade implica a preocupação com a segurança.

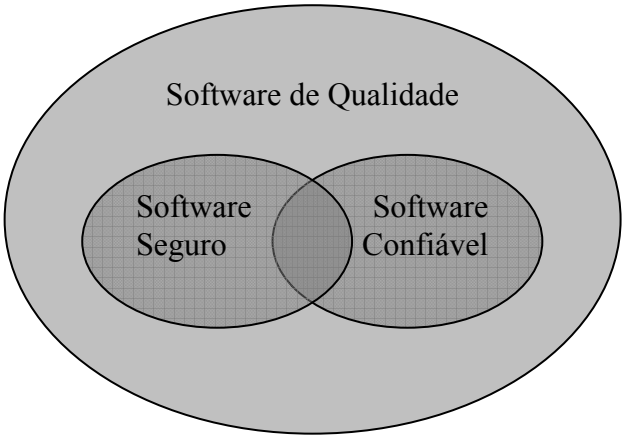

Figura 1 - *Software* seguro é um subconjunto do *software* de qualidade (HOWARD & LEBLANC, 2003).

2.2 ABRANGÊNCIA

2.2.1 Esta Instrução de Comando da Aeronáutica engloba as seguintes situações:

a) A aquisição, paga ou não, de *software* COTS;

b) O desenvolvimento e manutenção de *software* realizado por profissionais da própria organização; e

c) A contratação de terceiros para o desenvolvimento e manutenção de *software*.

2.2.2 A presente Norma inclui instruções para nortear as seguintes atividades:

a) Especificar requisitos técnicos de sistemas seguros;

b) Elaborar e seguir um processo de desenvolvimento seguro;

c) Auditar um processo de desenvolvimento seguro; e

d) Testar os requisitos técnicos de um sistema seguro.

2.2.3 Toda aquisição e desenvolvimento de sistemas de TI realizados no âmbito do DECEA e OM subordinadas devem atender a esta Norma.

2.3 PREMISSAS

2.3.1 A informação armazenada em sistemas e produtos de TI é um recurso crítico que propicia que as organizações sejam bem-sucedidas em sua missão (ISO/IEC 15408-1, 2009).

2.3.2 Grande parte das vulnerabilidades de segurança ocorre em consequência de ações realizadas durante o ciclo de desenvolvimento de um *software*.

2.3.3 A aplicabilidade dos requisitos técnicos da presente Norma deve ser norteada por uma análise de riscos e através de um direcionamento conduzido pela Política de Segurança da Organização.

2.3.4 O gerente de projetos responsável pela aquisição ou desenvolvimento do *software* deve fazer a seleção dos requisitos técnicos adequados, de acordo com o recomendado no item anterior.

2.3.5 A presente Instrução não depende de nenhuma metodologia de desenvolvimento ou modelo de ciclo de vida específicos.

3 RESPONSABILIDADES

3.1 SDTE – SUBDEPARTAMENTO TÉCNICO DO DECEA

3.1.1 Estabelecer normas, padrões e metodologias relativas a sistemas seguros, que estejam em conformidade com a legislação brasileira e com os padrões aceitos internacionalmente.

3.1.2 Estabelecer normas, padrões e metodologias que regularizem o emprego de controles específicos para sistemas seguros em sistemas da informação.

3.1.3 Realizar auditorias na execução dos processos de desenvolvimento de sistemas de informação no SISCEAB para garantir a conformidade com este documento normativo.

3.1.4 Realizar auditorias nos processos de desenvolvimento de sistemas de informação para avaliar a conformidade com este documento normativo.

3.1.5 Realizar análise de vulnerabilidades nos sistemas da informação em busca de vulnerabilidades que possam ser exploradas por um atacante, verificando os requisitos de sistemas seguros.

3.2 DESENVOLVEDOR DE SISTEMAS DE INFORMAÇÃO

3.2.1 Planejar o desenvolvimento do sistema de informação, usando como referência os requisitos técnicos de segurança da presente Norma.

3.2.2 Desenvolver o sistema de informação de acordo com os requisitos de segurança planejados, gerando uma matriz de rastreabilidade entre as funcionalidades implementadas e os requisitos estabelecidos.

3.2.3 Testar o sistema de informação para assegurar, constatando por intermédio da geração de evidências, os requisitos de segurança da informação implementados.

3.2.4 Realizar testes de vulnerabilidade dos sistemas desenvolvidos, preferencialmente usando uma ferramenta automática, e gerar relatórios de acordo com os anexos A e B da ICA 7-27.

3.2.5 Resolver ou mitigar as vulnerabilidades de alto impacto dos sistemas desenvolvidos antes de colocá-los em produção, registrando as ações de tratamento de acordo com o anexo C da ICA 7-27.

3.2.6 Identificar, quantificar e analisar as vulnerabilidades dos sistemas desenvolvidos, gerando registros de acordo com o anexo D da ICA 7-27.

3.3 CONTRATANTE DE SISTEMAS DE INFORMAÇÃO

3.3.1 Definir requisitos de segurança para o processo de desenvolvimento do sistema de informação, que porventura sejam aplicáveis, com base nesta Norma.

3.3.2 Realizar a seleção de sistemas de informação levando em consideração a aderência aos requisitos de segurança aqui definidos.

3.3.3 Garantir, por meio de evidências de testes, que o sistema de informação adquirido atende aos requisitos de segurança da informação definidos nesta Norma.

3.3.4 Realizar testes de vulnerabilidade dos sistemas adquiridos, preferencialmente usando uma ferramenta automática, e gerar relatórios de acordo com os anexos A e B da ICA 7-27.

3.3.5 Resolver ou mitigar as vulnerabilidades de alto impacto dos sistemas adquiridos, em conjunto com os fornecedores, antes de autorizar sua entrada em produção, registrando as ações de tratamento de acordo com o anexo C da ICA 7-27.

3.3.6 Identificar, quantificar e analisar as vulnerabilidades dos sistemas adquiridos, gerando registros de acordo com o anexo D da ICA 7-27.

4 DESENVOLVIMENTO DO SISTEMA

4.1 MELHORIAS NO PROCESSO DE DESENVOLVIMENTO

4.1.1 Para se construir um sistema seguro, devem ser realizadas melhorias de processo, priorizando os aspectos de segurança em cada fase do ciclo de vida de desenvolvimento do sistema, independentemente do modelo de ciclo de vida utilizado.

4.1.2 O SDL (*Security Development Lifecycle* – Ciclo de Vida do Desenvolvimento da Segurança) é um processo de desenvolvimento de *software* criado pela Microsoft em 2004 (MICROSOFT CORPORATION, 2010), que ajuda os desenvolvedores a construir *softwares* mais seguros e tratar os requisitos de conformidade, reduzindo simultaneamente o custo de desenvolvimento.

4.1.3 O SDL consiste em ações obrigatórias que seguem o processo tradicional de desenvolvimento de *software*, mas é flexível o suficiente para permitir a adição de outras políticas e técnicas, criando, assim, uma metodologia de desenvolvimento de *software* única para uma organização.

4.1.4 A presente ICA preconiza o emprego do SDL tanto na aquisição de *software* COTS quanto no desenvolvimento de sistemas de TI.

4.1.5 No caso de aquisição, deve ser realizada uma análise de conformidade que comprove a execução das práticas do SDL - ou de outras práticas que assegurem que as mesmas metas sejam alcançadas - para o sistema em questão.

4.1.6 No caso de desenvolvimento, devem ser elaborados requisitos de processo de forma a garantir que as práticas do SDL - ou outras práticas que assegurem que as mesmas metas sejam alcançadas - sejam seguidas.

4.1.7 Uma visão geral das etapas envolvidas no SDL é mostrada na figura abaixo. Cada uma das etapas será descrita mais adiante, em termos dos aspectos de segurança que nelas são contemplados.

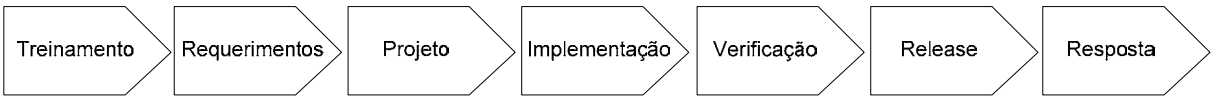

Figura 2 – Etapas do Processo de Desenvolvimento Seguro da Microsoft (MICROSOFT CORPORATION, 2010, p. 7, fig. 2)

4.1.8 Cada uma das etapas recomenda uma série de práticas para garantir sua implementação eficiente.

4.1.9 Como é possível notar da figura 2, a primeira etapa envolve treinamento. Ele é importante porque permite compreender os conceitos básicos de segurança e os mais recentes desenvolvimentos em segurança e privacidade, o que pode ajudar muito as organizações a reduzir o número e a gravidade das vulnerabilidades exploráveis do *software* e a reagir adequadamente aos cenários de ameaça em constante mudança.

4.1.9.1 O treinamento deve envolver os conceitos fundamentais para a construção de um *software* melhor, incluindo: *design* de segurança, modelagem de ameaças, desenvolvimento seguro, testes de segurança e melhores práticas relacionadas à privacidade.

4.1.10 A etapa seguinte trata de requerimentos. A fase de concepção do projeto é o melhor momento para considerar questões de privacidade e segurança fundamentais e analisar como alinhar a qualidade e os requisitos regulatórios com os custos e as necessidades de negócios.

4.1.10.1 As práticas desta fase são: "Estabelecer Requisitos de Segurança e Privacidade", "Especificar Quality Gates e Bug Bars" e "Realizar Avaliações de Risco de Segurança e Privacidade".

4.1.10.2 A prática "Estabelecer Requisitos de Segurança e Privacidade" recomenda definir e integrar os requisitos de segurança e privacidade logo de início. Isso ajuda a tornar mais fácil identificar os marcos e os entregáveis principais e a minimizar os desvios em planos e cronogramas. A análise de segurança e privacidade inclui a alocação de especialistas em segurança, a definição de critérios de privacidade e de segurança mínimos para um aplicativo e a implantação de um sistema de acompanhamento de vulnerabilidades de segurança e de itens de trabalho.

4.1.10.3 A prática "Especificar Quality Gates e Bug Bars" recomenda definir os níveis mínimos aceitáveis de qualidade de segurança e de privacidade no início do projeto. Isso ajuda uma equipe a compreender os riscos associados a problemas de segurança, identificar e a corrigir erros de segurança durante o desenvolvimento e a aplicar os padrões durante todo o projeto. Estabelecer uma bug bar representativa envolve definir claramente os limiares de severidade de vulnerabilidades de segurança (por exemplo, não deve haver vulnerabilidades conhecidas no aplicativo com uma classificação de "crítica" ou "importante" no momento da liberação), nunca permitindo que ceda, uma vez que tenha sido definida.

4.1.10.4 A prática "Realizar Avaliações de Risco de Segurança e Privacidade" recomenda examinar o design de software baseando-se em requisitos regulatórios e de custos. Isso auxilia uma equipe a identificar quais partes de um projeto exigirão a modelagem de ameaças e revisões de design de segurança antes do lançamento e a determinar a classificação de impacto de privacidade de um recurso, produto ou serviço.

4.1.11 A fase de projeto é fundamental para estabelecer as melhores práticas em torno do *design* e especificações funcionais e realizar análises de risco que ajudarão a atenuar as questões de segurança e privacidade ao longo de um empreendimento.

4.1.11.1 As práticas relativas a esta fase são: "Estabelecer Requisitos de Design", "Realizar Análise/Redução de Superfície de Ataque" e "Fazer Modelagem de Ameaças".

4.1.11.2 A prática "Estabelecer Requisitos de Design" aborda questões relativas à segurança e privacidade cedo. Isso ajuda a minimizar o risco de desvios de cronograma e a reduzir o custo de projetos. As especificações de design devem descrever os recursos de segurança e de privacidade que serão expostos diretamente para o usuário, tais como aqueles que requerem autenticação para acessar dados específicos ou conteúdos do usuário antes do uso de um recurso de privacidade de alto risco. Além disso, todas as especificações de design devem descrever como implementar toda a funcionalidade fornecida por um determinado recurso ou função de forma segura. Validar todas as especificações de design com relação à

especificação funcional envolve especificações de design precisas e completas, incluindo requisitos de design de criptografia mínimos e uma revisão da especificação.

4.1.11.3 A prática "Realizar Análise/Redução de Superfície de Ataque" aborda a redução das oportunidades de os atacantes explorarem um ponto fraco ou vulnerabilidade potenciais, o que requer analisar cuidadosamente a superfície de ataque global, e inclui desabilitar ou restringir o acesso aos serviços do sistema, aplicando o princípio de privilégio mínimo e empregando defesas em camadas, sempre que possível.

4.1.11.4 A prática "Fazer Modelagem de Ameaças" representa aplicar uma abordagem estruturada para os cenários de ameaças durante o projeto. Isso ajuda uma equipe mais eficazmente e menos dispendiosamente a identificar vulnerabilidades de segurança, determinar os riscos destas ameaças e estabelecer as mitigações adequadas.

4.1.12 O foco da fase de implementação é auxiliar o usuário final a tomar decisões informadas sobre as formas mais seguras para implantar o *software*. É também o momento de estabelecer as melhores práticas para detecção e remoção de problemas de segurança do código.

4.1.12.1 As práticas relativas a esta fase são: "Usar Ferramentas Aprovadas", "Proibir Funções Inseguras" e "Realizar Análise Estática".

4.1.12.2 A prática "Usar Ferramentas Aprovadas" diz respeito a publicar uma lista de ferramentas aprovadas e suas verificações de segurança associadas (tais como opções e os avisos do compilador/linker), o que ajuda a automatizar e reforçar as práticas de segurança facilmente a um baixo custo. Manter a lista regularmente atualizada significa que as últimas versões da ferramenta são usadas e permite a inclusão de novas funcionalidades e proteções de análise de segurança.

4.1.12.3 A prática "Proibir Funções Inseguras" refere-se a analisar todas as funções e API do projeto e vetar aquelas determinadas como inseguras, o que ajuda a reduzir possíveis bugs de segurança, com pouco custo de engenharia. As ações específicas incluem usar arquivos de cabeçalho (como "banned.h" e "strsafe.h"), compiladores mais atuais ou ferramentas de verficação de código para analisar o código com relação às funções na lista de proibidas e, em seguida, substituí-las por alternativas mais seguras.

4.1.12.4 A prática "Realizar Análise Estática" trata de analisar o código-fonte antes da compilação, fornecendo um método escalável de revisão do código de segurança e auxiliando a garantir que as políticas de codificação seguras estão sendo seguidas.

4.1.13 A fase de verificação envolve um esforço abrangente para garantir que o código atende aos princípios de segurança e de privacidade estabelecidos nas fases anteriores.

4.1.13.1 As práticas relativas a esta fase são: "Realizar Análise Dinâmica", "Realizar Testes de Fuzzing" e "Conduzir Revisão de Superfície de Ataque".

4.1.13.2 É importante garantir que a funcionalidade de um programa opere como planejado. A prática "Realizar Análise Dinâmica" desempenha a verificação em tempo de execução do software, checando a funcionalidade usando ferramentas que monitoram o comportamento do aplicativo quanto à corrupção de memória, problemas de privilégio do usuário e outros problemas de segurança críticos.

4.1.13.3 A prática “Realizar Testes de Fuzzing” induz deliberadamente à falha do programa introduzindo dados malformados ou aleatórios no aplicativo. Isso ajuda a revelar questões de segurança em potencial anteriormente ao lançamento, exigindo pouco investimento de recursos.

4.1.13.4 É comum para um aplicativo desviar significativamente das especificações funcionais e de design criadas durante as fases de requisitos e de design de um projeto de desenvolvimento de software. A prática “Conduzir Revisão de Superfície de Ataque” realiza a revisão da superfície de ataque após a conclusão do código, o que ajuda a garantir que quaisquer alterações de design ou implementação para um aplicativo ou sistema foram levadas em conta e que quaisquer novos vetores de ataque criados como resultado das alterações foram revistos e mitigados, incluindo modelos de ameaça.

4.1.14 O foco da fase de *Release* está na preparação de um projeto para liberação em produção, incluindo planejamento de maneiras para executar, efetivamente, tarefas de manutenção depois da liberação e abordar vulnerabilidades de segurança ou privacidade que podem ocorrer posteriormente. O objetivo final é garantir a segurança do *software* que será entregue ao cliente dentro das expectativas estabelecidas.

4.1.14.1 As práticas relativas a esta fase são: “Criar Plano de Resposta a Incidentes”, “Conduzir Revisão Final de Segurança” e “Certificar a Release e Arquivar”.

4.1.14.2 A prática “Criar Plano de Resposta a Incidentes” elabora um plano de resposta a incidentes, que é crucial para ajudar a tratar novas ameaças que podem surgir ao longo do tempo. Ele inclui a identificação de contatos de emergência de segurança adequados e estabelece planos de manutenção de segurança para código herdado de outros grupos dentro da organização e para código de terceiros licenciados.

4.1.14.3 A prática “Conduzir Revisão Final de Segurança” realiza a revisão de todas as atividades de segurança que foram feitas, o que ajuda a garantir a prontidão do lançamento do software. A Revisão Final de Segurança (FSR) geralmente inclui a análise de modelos de ameaças, das saídas de ferramentas e da análise de desempenho com os quality gates e bug bars definidos durante a fase de requisitos. A FSR redunda em um de três resultados diferentes: FSR aprovada, FSR aprovada com exceções ou FSR com escalação.

4.1.14.4 A prática “Certificar a Release e Arquivar” certifica o software antes de um lançamento, o que ajuda a garantir que os requisitos de segurança e de privacidade foram atendidos. Arquivar todos os dados pertinentes é essencial para a realização de tarefas de manutenção depois do lançamento e ajuda a reduzir os custos a longo prazo associados com a engenharia de software sustentável. O arquivamento deve incluir todas as especificações, código-fonte, binários, símbolos privados, modelos de ameaças, documentação, planos de resposta de emergência, e termos de licença e de serviços para qualquer software de terceiros.

4.1.15 A fase de Resposta centra-se na equipe de desenvolvimento estar apta e disponível para responder adequadamente a quaisquer relatórios de vulnerabilidades e ameaças emergentes de *software*.

4.1.15.1 Esta fase possui uma prática, “Executar Plano de Resposta a Incidentes”, que trata de ser capaz de implementar o plano de resposta a incidentes instituído na fase de release, o que é essencial para ajudar a proteger os usuários contra vulnerabilidades de segurança ou de privacidade que surjam no software.

4.2 SEGURANÇA POR CÓDIGO

4.2.1 A segurança por código diz respeito a técnicas para tratar vulnerabilidades e cuidados que devem ser tomados ao se desenvolver *software* (CLAVIS SEGURANÇA DA INFORMAÇÃO, 2013, *slides* 127 a 161).

4.2.2 As práticas recomendadas nesta seção atendem às técnicas básicas preconizadas pelo OWASP *Top* 10 para se proteger contra as mais importantes vulnerabilidades de segurança de aplicações *Web*.

4.2.3 A presente ICA aconselha o emprego da segurança por código tanto na aquisição de *software* COTS quanto no desenvolvimento de sistemas de TI.

4.2.4 No caso de aquisição, deve ser realizada uma análise de conformidade que comprove a implementação das técnicas de segurança por código para o sistema em questão.

4.2.5 No caso de desenvolvimento, devem ser elaborados requisitos de processo de forma a garantir que as técnicas de segurança por código sejam seguidas.

4.2.6 PRÁTICAS GERAIS DE CÓDIGO SEGURO

4.2.6.1 Não se deve armazenar senhas em código-fonte.

4.2.6.2 Não se deve utilizar códigos da Internet sem conhecer a fonte ou entender seu funcionamento.

4.2.6.3 Não se deve usar funções ou recursos obsoletos (*deprecated*).

4.2.6.4 Deve-se aplicar o conceito de validação positiva.

4.2.7 VALIDAÇÃO DE ENTRADAS E CODIFICAÇÃO DE SAÍDA

4.2.7.1 Todo ponto de interação de dados com o usuário do sistema (*input*) deve ter sua validação feita na entrada do dado e na sua apresentação (*output*).

4.2.7.2 Validações de segurança devem ser realizadas no servidor (*Webserver*) sempre.

4.2.7.3 No cliente (navegador), a validação pode ser feita quando convier.

4.2.7.4 Sempre que possível, deve-se adotar a validação positiva ou lista branca (*whitelist*).

4.2.7.5 Sempre se deve pensar na possibilidade de *bypass* da validação.

4.2.8 AUTENTICAÇÃO E GERENCIAMENTO DE SENHAS

4.2.8.1 Deve-se utilizar sempre HTTP POST para a requisição de autenticação.

4.2.8.2 Deve-se utilizar HTTPS para envio de credenciais.

4.2.8.3 Deve-se pedir sempre uma nova autenticação em ações críticas.

4.2.8.4 Deve-se usar *hash* com *salt* para codificar as senhas a serem armazenadas.

4.2.8.5 Deve-se usar um algoritmo de *hash* público e aprovado por instituto de padronização de renome internacional como, por exemplo, o *National Institute of Standards and Technology* (NIST) dos Estados Unidos. Não se deve usar algoritmos de hash considerados fracos pela maioria da comunidade técnica internacional.

4.2.8.6 Deve-se implementar complexidade de senhas e não permitir usuários ou senhas-padrão. Para tal, toda aplicação que exija *login* e senha deve usar uma classe ou mecanismo validador de senhas para garantir o cumprimento da política de senhas do DECEA.

4.2.8.7 Deve-se desabilitar as opções de configuração "lembrar minha senha" e o "autocompletar" de navegadores.

4.2.8.8 Deve-se usar mensagens imprecisas para informar uma tentativa de *login* inválida. Usar: "usuário ou senha inválidos". Não usar: "usuário inexistente" ou "senha incorreta".

4.2.8.9 Deve-se planejar bem a lógica dos processos de "esqueci a minha senha".

4.2.9 AUTORIZAÇÃO E GERENCIAMENTO DE ACESSO

4.2.9.1 A restrição de acesso para cada perfil a um conteúdo ou operação específicos não deve ser feita apenas pela ocultação de um *link* ou de uma opção em um *menu*.

4.2.9.2 Deve-se fazer uma validação de cada requisição, sempre no servidor.

4.2.9.3 Deve-se garantir que arquivos e diretórios não possam ser acessados diretamente.

4.2.9.4 Não se deve armazenar nenhuma informação que possa comprometer o controle de acesso do lado do cliente.

4.2.9.5 Deve-se forçar um *re-login* sempre que o perfil de usuário for alterado, desta forma um usuário não pode permanecer com o acesso caso tenha sofrido um *downgrade* de perfil.

4.2.10 GERENCIAMENTO DE SESSÃO

4.2.10.1 O controle de sessão deve ser tratado pelo servidor, apenas a persistência dele no cliente.

4.2.10.2 Deve-se implementar *time-out* de sessão e sempre gerar um novo ID após a autenticação.

4.2.10.3 Não se deve passar ID de sessão por HTTP GET.

4.2.10.4 Deve-se implementar um *token* de sessão por requisição e com alta aleatoriedade para evitar ataques de CSRF.

4.2.11 TRANSMISSÃO E ARMAZENAMENTO DE INFORMAÇÕES SENSÍVEIS

4.2.11.1 Dados sensíveis devem ser transmitidos por canais HTTPS.

4.2.11.2 Não se deve armazenar senhas ou dados sensíveis em locais sem criptografia.

4.2.11.3 Não se deve enviar informações sensíveis para *logs*.

4.2.11.4 Deve-se certificar que os comentários de código não contenham informações que possam comprometer a segurança.

4.2.11.5 Deve-se garantir que a lógica do código do lado do cliente não crie vulnerabilidades.

4.2.11.6 Campos *hidden* não devem armazenar dados sensíveis.

4.2.11.7 Deve-se estudar a adoção do cabeçalho HSTS (HTTP *Strict Transport Security*)

4.2.11.8 Campos *password* podem ser revertidos do lado do cliente, portanto não se deve apresentar dados sensíveis usando esse recurso.

4.2.12 INTERAÇÃO COM BANCO DE DADOS

4.2.12.1 Sempre que possível, deve-se adotar *Stored Procedures* ou *Prepared Statement*.

4.2.12.2 Se não for possível usar *Stored Procedures* ou *Prepared Statement,* deve-se validar todas as interações com o banco e nunca se deve usar concatenação para montar qualquer consulta com o banco de dados.

4.2.12.3 Deve-se desabilitar os recursos do banco que não serão utilizados.

4.2.12.4 Deve-se acessar o banco de dados com um usuário com privilégio estritamente necessário para a realização das operações necessárias.

4.2.12.5 Não se deve deixar o banco de dados ser acessado por outro canal que não seja a aplicação.

4.2.13 GERENCIAMENTO DE ARQUIVOS

4.2.13.1 Deve-se garantir que os arquivos de configuração, documentação e diagnósticos não estejam publicados.

4.2.13.2 Não devem existir arquivos de *backup* e arquivos antigos na área de execução do *site.*

4.2.13.3 Não se deve usar arquivos com extensões que não são processadas pelo servidor de aplicação. Por exemplo: .inc, .class, .txt.

4.2.14 TRATAMENTO ADEQUADO DE ERROS

4.2.14.1 Deve-se usar uma estrutura de tratamento de erros em qualquer método em que um erro possa ocorrer.

4.2.14.2 O Java não encerra as conexões de banco de dados, arquivos etc. quando erros ocorrem. Nesse caso, deve-se usar, portanto, o bloco *finally* para realizar as ações de limpeza necessárias.

4.2.15 *LOGGING*

4.2.15.1 O *logging* deve fazer parte da concepção do sistema, deve estar claro o que se pretende com os *logs* da aplicação.

4.2.15.2 Deve-se implementar um bom sistema de *logging*, de tal forma a viabilizar auditorias e investigações de fraudes e casos de abuso.

4.3 REQUISITOS DE SEGURANÇA

4.3.1 As seguintes categorias de requisitos devem ser especificadas para a aquisição ou desenvolvimento de um sistema seguro:

a) Requisitos de identificação: especificam a extensão em que um aplicativo ou componente identifica entidades externas (por exemplo, atores humanos e aplicações externas) antes de interagir com eles;

b) Requisitos de autenticação: especificam o grau em que as identidades de fontes externas são verificadas antes de permitir-lhes solicitar e receber serviços (por exemplo, executar funções, obter dados);

c) Requisitos de autorização: especificam o grau em que um aplicativo ou componente corretamente concede e reforça os privilégios de acesso e de uso de entidades externas autenticadas;

d) Requisitos de auditoria de segurança: especificam o grau em que a equipe de segurança é habilitada a auditar o *status* e o uso de mecanismos de segurança através da análise de eventos relacionados à segurança;

e) Requisitos de confidencialidade: especificam a extensão em que um aplicativo ou componente deve assegurar que seus dados e comunicações não são expostos a usuários não autorizados;

f) Requisitos de integridade: são requisitos de segurança que especificam a extensão a que um aplicativo ou componente deve assegurar que seus dados e comunicações não são intencionalmente corrompidos através de criação, modificação ou exclusão não autorizadas;

g) Requisitos de disponibilidade: especificam a medida que um aplicativo ou componente deve se manter operando e estar disponível para todos os seus usuários autorizados;

h) Requisitos de irretratabilidade: especificam o grau em que um participante de uma interação (por exemplo, mensagem, transação, transmissão de dados) é impedido de negar com êxito qualquer aspecto desta interação;

i) Requisitos de imunidade: especificam o grau em que o sistema se protege de infecção por programas maliciosos não autorizados (por exemplo, vírus, *worms*, cavalos de Troia, *scripts* maliciosos);

j) Requisitos de capacidade de sobrevivência: são requisitos de segurança que especificam a medida em que um aplicativo ou componente deve sobreviver à perda ou à destruição de partes significativas, causadas por ataque ou acidente;

k) Requisitos de segurança de manutenção de sistemas: especificam a extensão em que o sistema impede que modificações autorizadas durante a manutenção acidentalmente debilitem seus mecanismos de segurança; e

l) Requisitos de privacidade: especificam a extensão em que um aplicativo ou componente deve fornecer provisões para exercer os direitos e obrigações

dos indivíduos e organizações com relação a coleta, uso, retenção, divulgação e eliminação de informações pessoais.

4.4 CASOS DE TESTES DE SEGURANÇA

4.4.1 Os casos de teste descrevem exatamente o que será executado e quais são as funcionalidades do sistema de TI que estão sendo verificadas. No caso de teste de segurança, são testadas as funcionalidades e requisitos relacionados à segurança do sistema.

4.4.2 Cada requisito de segurança descrito no item 4.2.1 deve ser vinculado a um ou mais casos de teste.

4.4.3 Esses casos de teste devem ser definidos quando se estiver desenvolvendo um Plano de Testes, durante a fase de análise do projeto. Esse Plano deve ser empregado para testes completos, tanto dos requisitos funcionais da aplicação como de seus requisitos de segurança.

4.5 REQUISITOS MÍNIMOS DE SEGURANÇA DA INFORMAÇÃO PARA SISTEMAS CRÍTICOS

(ABNT NBR ISO/IEC 27002, 2005, cap. 12)

Deve-se garantir que a segurança da informação é parte integrante de sistemas críticos.

4.5.1 ANÁLISE E ESPECIFICAÇÃO DOS REQUISITOS DE SEGURANÇA DA INFORMAÇÃO

4.5.1.1 As especificações para os requisitos de controles, nos sistemas críticos, devem considerar os controles automáticos a serem incorporados, assim como a necessidade de apoiar controles manuais. Considerações similares devem ser aplicadas na avaliação de pacotes de *softwares*, desenvolvidos internamente ou adquiridos, para os sistemas críticos em operação no DECEA.

4.5.1.2 Os requisitos de segurança da informação e os controles de segurança da informação devem refletir o valor para a organização dos ativos de informação envolvidos e os danos potenciais à organização que poderiam resultar de uma falha ou ausência de segurança da informação.

4.5.1.3 Os requisitos de sistemas críticos para a segurança da informação, bem como os processos para implementá-los, devem ser integrados aos estágios iniciais dos projetos dos sistemas críticos. Controles introduzidos no estágio de projeto são significativamente menos dispendiosos para implementar e manter do que aqueles incluídos durante ou após a implementação.

4.5.1.4 No caso de produtos adquiridos, os requisitos de segurança presentes nesta Norma devem ser incluídos em contrato e os testes aqui recomendados devem ser realizados.

4.5.1.5 Os contratos com fornecedores devem levar em consideração os requisitos de segurança da informação identificados. Nas situações em que funcionalidades de segurança da informação de um produto proposto não satisfaçam requisitos especificados, o risco introduzido, assim como os controles associados, devem ser reconsiderados antes da compra do produto. Nas situações em que as funcionalidades adicionais incorporadas acarretem riscos

à segurança, estas funcionalidades devem ser desativadas ou a estrutura de controles proposta deve ser analisada criticamente para determinar se há vantagem na utilização das funcionalidades em questão.

4.5.1.6 Os controles que seguem se correlacionam com as categorias de requisitos descritas no item 4.2 de acordo com a tabela abaixo:

GRUPO DE CONTROLES	CONTROLE DE SISTEMAS CRÍTICOS	CATEGORIA DE REQUISITOS
Processamento Correto nos Sistemas Críticos	Validação dos dados de entrada	Requisitos de integridade
	Controle do processamento interno	Requisitos de integridade
	Integridade de mensagens	Requisitos de integridade
	Validação de dados de saída	Requisitos de integridade
Controles Criptográficos	Controles criptográficos	Requisitos de confidencialidade, Requisitos de irretratabilidade
Segurança dos Arquivos do Sistema	Controle de *software* operacional	Requisitos de integridade
	Proteção dos dados para teste de sistema	Requisitos de confidencialidade
	Controle de acesso ao código-fonte de programa	Requisitos de autorização
Segurança em Processos de Desenvolvimento e Suporte	Procedimentos para controle de mudanças	Requisitos de segurança de manutenção de sistemas
	Análise crítica técnica dos sistemas após mudanças no sistema operacional	Requisitos de segurança de manutenção de sistemas
	Restrições sobre mudanças em pacotes de *software*	Requisitos de segurança de manutenção de sistemas
	Vazamento de informações	Requisitos de confidencialidade
	Desenvolvimento terceirizado de *software*	Requisitos de auditoria de segurança
Gestão de Vulnerabilidades Técnicas	Controle de vulnerabilidades técnicas	Requisitos de auditoria de segurança, Requisitos de segurança de manutenção de sistemas, Requisitos de disponibilidade

4.5.2 PROCESSAMENTO CORRETO NOS SISTEMAS CRÍTICOS

4.5.2.1 Introdução

4.5.2.1.1 Deve-se prevenir a ocorrência de erros, perdas, modificação não autorizada ou uso inadequado de informações em sistemas críticos.

4.5.2.1.2 Controles apropriados devem ser incorporados no projeto das aplicações para assegurar o processamento correto. Os controles devem incluir a validação dos dados de

entrada, do processamento interno e dos dados de saída.

4.5.2.2 Validação dos dados de entrada

4.5.2.2.1 Devem ser aplicadas checagens na entrada de transações da organização, em dados permanentes (por exemplo, nomes e endereços, limites de crédito e números de referência de clientes) e em tabelas de parâmetros (por exemplo, preços de venda, taxas de conversão de moedas e tarifas de impostos).

4.5.2.2.2 Devem ser consideradas as seguintes diretrizes de segurança da informação:

a) verificação de entrada duplicada ou outros tipos de verificação, tais como checagem de limites ou campos limitando as faixas específicas de dados de entrada, para detectar os seguintes erros:
   - valores fora de faixa;
   - caracteres inválidos em campos de dados;
   - dados incompletos ou faltantes;
   - volumes de dados excedendo limites superiores ou inferiores; e
   - dados de controle inconsistentes ou não autorizados.

b) verificação periódica do conteúdo de campos-chave ou arquivos de dados para confirmar a sua validade e integridade;

c) inspeção de cópias impressas de documentos de entrada para detectar quaisquer alterações não autorizadas. Todas as mudanças em documentos de entrada devem ser autorizadas;

d) criação de algoritmos para tratar erros de validação;

e) criação de algoritmos para testar a credibilidade dos dados de entrada;

f) definição das responsabilidades de toda a equipe envolvida no processo de entrada de dados; e

g) criação de um registro de atividades envolvendo o processo de entrada de dados.

4.5.2.3 Controle do processamento interno

4.5.2.3.1 O projeto e a implementação dos sistemas críticos devem garantir que os riscos de falhas de processamento que acarretem a perda de integridade sejam minimizados. Cuidados específicos a serem tomados incluem:

a) o uso das funções, como incluir, modificar e remover para implementação de alterações nos dados;

b) procedimentos para evitar que programas executem suas instruções na ordem errada ou continuem em execução após uma falha de processamento;

c) o uso de programas apropriados para recuperação de falhas, para assegurar o processamento correto dos dados; e

d) inclusão de proteção contra a ocorrência de *buffer overrun/overflow*.

4.5.2.3.2 Devem ser elaboradas listas de verificação adequadas, documentar as atividades e manter os resultados em segurança. Exemplos de verificações que podem ser incorporadas incluem:

a) controles de seções ou de lotes, para reconciliar saldos de arquivos após as atualizações de transações;

b) controles de saldos, para verificação de saldos abertos comparando com saldos previamente encerrados o batimento de saldos de abertura contra saldos de fechamento, utilizando totalizações na atualização de arquivos;

c) validação de dados de entrada gerados pelo sistema;

d) verificações de integridade, autenticidade ou qualquer outra característica de segurança, de dados ou *softwares* transferidos, ou atualizados entre computadores centrais e remotos;

e) implementação de técnicas de consistência (*hash*) para registros e arquivos;

f) verificações para garantir que os programas sejam executados no tempo correto;

g) verificações para garantir que os programas sejam executados na ordem correta e terminem em caso de falha, e que qualquer processamento adicional seja suspenso, até que o problema seja resolvido; e

h) criação de um registro das atividades envolvidas no processamento.

4.5.2.4 Integridade de mensagens

Deve ser executada uma análise/avaliação dos riscos de segurança (ICA 7-26, 2013) para determinar se a integridade das mensagens é requerida e para identificar o método mais apropriado de implementação.

4.5.2.5 Validação de dados de saída

Os dados de saída dos sistemas críticos devem ser validados para assegurar que o processamento das informações armazenadas está correto e é apropriado às circunstâncias.

4.5.2.5.1 A validação de dados de saída deve incluir:

a) verificações de credibilidade para testar se os dados de saída são razoáveis;

b) controles envolvendo contadores de reconciliação para garantir o processamento de todos os dados;

c) fornecimento de informação suficiente para que um leitor ou um sistema de processamento subsequente possa determinar a exatidão, completude, precisão e classificação das informações;

d) medidas para aplicar em resposta aos testes de validação dos dados de saída;

e) definição das responsabilidades de toda a equipe envolvida no processo de dados de saída; e

f) criação de um registro de atividades do processo de validação dos dados de saída.

4.5.3 CONTROLES CRIPTOGRÁFICOS

4.5.3.1 Um procedimento de segurança da informação deve ser desenvolvido para o uso de controles criptográficos.

4.5.3.2 O gerenciamento de chaves deve ser implementado para apoiar o uso de técnicas criptográficas.

4.5.4 SEGURANÇA DOS ARQUIVOS DO SISTEMA

Deve-se garantir a segurança da informação de arquivos de sistema. O acesso aos arquivos de sistema e aos programas de código-fonte devem ser controlados e as atividades de projeto de Tecnologia de Informação e de suporte devem ser conduzidas de forma segura. Cuidados devem ser tomados para evitar a exposição de dados sensíveis em ambiente de teste.

4.5.4.1 Controle de *software* operacional

4.5.4.1.1 Para minimizar o risco de corrupção desses sistemas, devem ser observados os seguintes requisitos para controlar mudanças:

a) a atualização do *software* operacional, de aplicativos e de bibliotecas de programas deve ser executada somente por administradores treinados e com autorização gerencial;

b) *softwares* operacionais devem conter somente código executável e aprovado, e não devem conter códigos em desenvolvimento ou compiladores;

c) sistemas operacionais e aplicativos somente devem ser implementados após testes extensivos e bem-sucedidos; é recomendável que os testes incluam testes sobre usabilidade, segurança, efeitos sobre outros sistemas, como também sobre uso amigável, e sejam realizados em sistemas separados; deve ser assegurado que todas as bibliotecas de programa-fonte correspondentes tenham sido atualizadas;

d) um sistema de controle de configuração deve ser utilizado para manter controle da implementação do *software*, assim como da documentação do sistema;

e) uma estratégia de retorno (*rollback*) às condições anteriores deve ser disponibilizada antes que mudanças sejam implementadas no sistema;

f) um registro de auditoria deve ser mantido para todas as atualizações das bibliotecas dos programas operacionais;

g) versões anteriores dos *softwares* aplicativos devem ser mantidas como medida de contingência; e

h) versões antigas de *software* devem ser arquivadas, junto com todas as informações e parâmetros requeridos, procedimentos, detalhes de configurações e *software* de suporte durante um prazo igual ao prazo de retenção dos dados.

4.5.4.1.2 O sistema crítico adquirido de fornecedores e utilizado em sistemas envolvidos com a operação deve ser atualizado, de tal forma a mantê-lo em um nível suportado pelo

fornecedor. No decorrer do tempo, os fornecedores de sistema cessam o apoio às versões antigas do *software*. Caso não contrate a atualização desses sistemas, o contratante de sistemas de informação deve avaliar os riscos inerentes à dependência de sistemas sem suporte.

4.5.4.1.3 Qualquer decisão de atualização para uma nova versão deve considerar os requisitos do negócio para a mudança e da segurança da informação associada, por exemplo, a introdução de uma nova funcionalidade de segurança ou a quantidade e a gravidade dos problemas de segurança associados a esta versão. Pacotes de correções de *software* devem ser aplicados quando puderem remover ou reduzir as vulnerabilidades de segurança.

4.5.4.1.4 Os acessos físicos e lógicos devem ser concedidos a fornecedores, quando necessário, para a finalidade de suporte e com aprovação gerencial. Todas as atividades do fornecedor devem ser monitoradas.

4.5.4.1.5 Os sistemas operacionais devem ser atualizados quando existir um requisito para tal, por exemplo, se a versão atual do sistema operacional não suportar mais os requisitos do negócio. As atualizações não devem ser efetivadas pela simples disponibilidade de uma versão nova do sistema operacional. Novas versões de sistemas operacionais podem ser menos seguras, com menor estabilidade e ser menos entendidas do que os sistemas atuais.

4.5.4.2 Proteção dos dados para teste de sistema

4.5.4.2.1 Para propósitos de teste, deve ser evitado o uso de bancos de dados operacionais que contenham informações de natureza pessoal ou qualquer outra informação considerada sensível. Se informação de natureza pessoal ou outras informações sensíveis forem utilizadas com o propósito de teste, todos os detalhes e conteúdo sensível devem ser removidos ou modificados de forma a evitar reconhecimento antes do seu uso.

4.5.4.2.2 Devem ser aplicados os seguintes requisitos de segurança da informação para a proteção de dados operacionais, quando utilizados para fins de teste:

a) os procedimentos de controle de acesso, aplicáveis aos aplicativos de sistema em ambiente operacional, devem também ser aplicados aos aplicativos de sistema em ambiente de teste;

b) seja obtida autorização cada vez que for utilizada uma cópia da informação operacional para uso de um aplicativo em teste;

c) a informação operacional seja apagada do aplicativo em teste imediatamente após completar o teste; e

d) a cópia e o uso de informação operacional sejam registrados de forma a prover uma trilha para auditoria.

4.5.4.3 Controle de acesso ao código-fonte de programa

4.5.4.3.1 O acesso ao código-fonte de programa e de itens associados (como desenhos, especificações, planos de verificação e de validação) deve ser estritamente controlado, com a finalidade de prevenir a introdução de funcionalidades não autorizadas e para evitar mudanças não intencionais.

4.5.4.3.2 Para os códigos-fonte de sistemas, este controle pode ser obtido com a guarda centralizada do código, de preferência utilizando bibliotecas de programa-fonte.

4.5.4.3.3 Os seguintes requisitos de segurança da informação devem ser considerados para o controle de acesso às bibliotecas de programa-fonte, com a finalidade de reduzir o risco de corrupção de programas de computador:

a) quando possível, seja evitado manter as bibliotecas de programa-fonte no mesmo ambiente dos sistemas operacionais;

b) seja implementado o controle do código-fonte de programa e das bibliotecas de programa-fonte, conforme procedimentos estabelecidos;

c) o pessoal de suporte não tenha acesso irrestrito às bibliotecas de programa-fonte;

d) a atualização das bibliotecas de programa-fonte e itens associados e a entrega de fontes de programas a programadores devem ser efetuadas após o recebimento da autorização pertinente;

e) as listagens dos programas sejam mantidas em um ambiente seguro;

f) seja mantido um registro de auditoria de todos os acessos a código-fonte de programa; e

g) a manutenção e a cópia das bibliotecas de programa-fonte estejam sujeitas a procedimentos estritos de controles de mudanças.

4.5.5 SEGURANÇA EM PROCESSOS DE DESENVOLVIMENTO E DE SUPORTE

Deve-se manter a segurança de sistemas críticos e da informação. Os gerentes responsáveis pela segurança dos ambientes de projeto ou de suporte devem assegurar que quaisquer mudanças propostas sejam analisadas criticamente para verificar que não comprometem a segurança do sistema ou do ambiente operacional.

4.5.6 PROCEDIMENTOS PARA CONTROLE DE MUDANÇAS

4.5.6.1 Os procedimentos de segurança da informação de controle de mudanças devem ser documentados e reforçados com a finalidade de minimizar a corrupção dos sistemas críticos.

4.5.6.2 A introdução de novos sistemas e mudanças maiores em sistemas existentes devem seguir um processo formal de documentação, especificação, teste, controle da qualidade e gestão da implementação.

4.5.6.3 O processo deve incluir uma análise/avaliação de riscos, análise do impacto das mudanças e a especificação dos controles de segurança da informação requeridos – para tal recomenda-se o uso da metodologia da norma ISO/IEC 27005. O processo deve garantir que a segurança da informação e os procedimentos de controle atuais não sejam comprometidos, que os programadores de suporte tenham acesso somente às partes do sistema necessárias ao cumprimento das tarefas e que sejam obtidas concordância e aprovação formal para qualquer mudança obtida.

4.5.6.4 Quando praticável, os procedimentos de segurança da informação de controle de mudanças devem ser integrados.

4.5.6.5 Os procedimentos de segurança da informação para sistemas críticos devem incluir os seguintes requisitos de segurança da informação abaixo:

a) a manutenção de um registro dos níveis acordados de autorização;

b) a garantia de que as mudanças sejam submetidas por usuários autorizados;

c) a análise crítica dos procedimentos de controle e integridade para assegurar que as mudanças não os comprometam;

d) a identificação de todo sistema, informação, entidades em bancos de dados e *hardware* que precisam de emendas;

e) a obtenção de aprovação formal para propostas detalhadas antes da implementação;

f) a garantia da aceitação das mudanças por usuários autorizados, antes da implementação;

g) a garantia da atualização da documentação do sistema após conclusão de cada mudança e de que a documentação antiga seja arquivada ou descartada;

h) a manutenção de um controle de versão de todas as atualizações de sistemas críticos;

i) a manutenção de uma trilha para auditoria de todas as mudanças solicitadas;

j) a garantia de que toda a documentação operacional e procedimentos dos usuários sejam alterados conforme necessário e que se mantenham apropriados; e

k) a garantia de que as mudanças sejam implementadas em horários apropriados, sem interferir nos processos operacionais regulares.

4.5.7 ANÁLISE CRÍTICA TÉCNICA DOS SISTEMAS APÓS MUDANÇAS NO SISTEMA OPERACIONAL

4.5.7.1 Quando houver mudanças nos sistemas operacionais, as aplicações críticas devem ser revisadas e testadas para garantir que não haverá nenhum impacto adverso na operação da organização ou na segurança.

4.5.7.2 O processo de análise crítica técnica dos sistemas após mudanças no sistema operacional deve compreender os seguintes requisitos de segurança da informação abaixo:

a) uma análise crítica dos procedimentos de controle e integridade dos controles para assegurar que não foram comprometidos pelas mudanças no sistema operacional;

b) a garantia de que o plano anual de suporte e o orçamento irão cobrir as análises e testes do sistema devido às mudanças no sistema operacional;

c) a garantia de que as mudanças pretendidas sejam comunicadas em tempo hábil para permitir os testes e análises críticas antes da implementação das mudanças; e

d) a garantia de que as mudanças necessárias sejam executadas nos planos de continuidade de negócios.

4.5.7.3 Deve ser dada responsabilidade a um grupo específico ou a um indivíduo para monitoramento das vulnerabilidades e divulgação de emendas e correções dos fornecedores de sistemas.

4.5.8 RESTRIÇÕES SOBRE MUDANÇAS EM PACOTES DE *SOFTWARE*

4.5.8.1 Quando possível e praticável, que pacotes de *softwares* providos pelos fornecedores sejam utilizados sem modificações. Quando um pacote de *software* requer modificação, devem ser considerados os seguintes requisitos de segurança da informação:

a) o risco de que controles e processos de integridade contidos no *software* sejam comprometidos;

b) a obtenção do consentimento do fornecedor;

c) a possibilidade de obtenção junto ao fornecedor das mudanças necessárias para a atualização padrão do programa; e

d) o impacto resultante quando a organização passa a ser responsável pela manutenção futura do *software* como resultado das mudanças.

4.5.8.2 Se mudanças forem necessárias, o *software* original deve ser mantido e as mudanças aplicadas numa cópia claramente identificada. O processo de gestão de atualizações deve ser implementado para assegurar a instalação das mais recentes correções e atualizações para todos os *softwares* autorizados.

4.5.8.3 Todas as mudanças devem ser completamente testadas e documentadas para que possam ser reaplicadas, se necessário, em atualizações futuras do *software*. Se requerido, as modificações devem ser testadas e validadas por um grupo de avaliação independente.

4.5.8.4 O processo de gestão de mudanças do DECEA é descrito em Instrução específica (ICA 7-24, 2013).

4.5.9 VAZAMENTO DE INFORMAÇÕES

4.5.9.1 Os seguintes itens abaixo devem ser considerados, para limitar o risco de vazamento de informações, por exemplo, através do uso e exploração de *covert channels*:

a) a varredura do envio de mídia e comunicações para verificar a presença de informação oculta;

b) o mascaramento e a modulação do comportamento dos sistemas e das comunicações para reduzir a possibilidade de terceiros deduzirem informações a partir do comportamento dos sistemas;

c) a utilização de sistemas e *software* reconhecidos como de alta integridade, por exemplo, utilizando produtos avaliados (ver ISO/IEC 15408-1);

d) o monitoramento regular das atividades do pessoal e do sistema, quando permitido pela legislação ou regulamentação vigente; e

e) o monitoramento do uso de recursos de sistemas de computação.

4.5.10 DESENVOLVIMENTO TERCEIRIZADO DE *SOFTWARE*

Devem ser considerados os seguintes requisitos de segurança da informação, quando do desenvolvimento de sistemas terceirizados:

a) acordos de licenciamento, propriedade do código e direitos de propriedade intelectual;

b) certificação da qualidade e exatidão do serviço realizado;

c) provisões para custódia no caso de falha de terceiros;

d) direitos de acesso para auditorias de qualidade e exatidão do serviço realizado;

e) requisitos contratuais para a qualidade e funcionalidade da segurança do código; e

f) testes antes da instalação para detectar a presença de código malicioso e cavalos de Troia.

4.5.11 GESTÃO DE VULNERABILIDADES TÉCNICAS

Devem-se reduzir riscos resultantes da exploração de vulnerabilidades técnicas conhecidas mediante a Gestão de Vulnerabilidades Técnicas (ICA 7-27, 2013).

4.5.11.1 Controle de vulnerabilidades técnicas

4.5.11.1.1 Um inventário completo e atualizado dos ativos de informação é um pré-requisito para uma gestão efetiva de vulnerabilidade técnica. Informação específica para o apoio à gestão de vulnerabilidade técnica inclui o fornecedor de sistemas críticos, o número de versão, o *status* atual de uso e distribuição (por exemplo, que *softwares* estão instalados e em quais sistemas) e a(s) pessoa(s) na organização responsável(is) pelos *softwares*.

4.5.11.1.2 A ação apropriada deve ser tomada, no devido tempo, como resposta às potenciais vulnerabilidades técnicas identificadas. As seguintes diretrizes devem ser seguidas para o estabelecimento de um processo de gestão efetivo de vulnerabilidades técnicas:

a) O DECEA deve estabelecer as funções e responsabilidades associadas à gestão de vulnerabilidades técnicas, incluindo o monitoramento de vulnerabilidades, a análise/avaliação de riscos de vulnerabilidades, *patches*, acompanhamento dos ativos e qualquer coordenação de responsabilidades requerida (ICA 7-27, 2013; item 3);

b) os recursos de informação a serem utilizados para identificar vulnerabilidades técnicas relevantes e para manter a conscientização sobre elas devem ser identificados para *softwares* e outras tecnologias (com base na lista de inventário dos ativos); esses recursos de informação devem ser mantidos atualizados com base nas mudanças no inventário de ativos, ou quando outros recursos novos ou úteis forem encontrados;

c) seja definido um prazo para a reação a notificações de potenciais vulnerabilidades técnicas relevantes;

d) uma vez que uma vulnerabilidade técnica potencial tenha sido identificada, a organização deve avaliar os riscos associados e as ações a serem tomadas; tais ações podem requerer o uso de *patches* nos sistemas vulneráveis e/ou a aplicação de outros controles;

e) dependendo da urgência exigida para tratar uma vulnerabilidade técnica, a ação tomada deve estar de acordo com os controles relacionados à gestão de

mudanças ou devem ser seguidos os procedimentos de resposta a incidentes de segurança da informação (ICA 7-23, 2013);

f) se um *patch* for disponibilizado, devem ser avaliados os riscos associados à sua instalação (os riscos associados à vulnerabilidade devem ser comparados com os riscos de instalação do *patch*) (ICA 7-26, 2013; item 3.4);

g) *patches* devem ser testados e avaliados antes de serem instalados para assegurar a efetividade e de modo que não tragam efeitos que não possam ser tolerados; quando não existir a disponibilidade de um *patch*, deve-se considerar o uso de outros controles, tais como:
   - a desativação de serviços ou potencialidades relacionadas à vulnerabilidade;
   - a adaptação ou agregação de controles de acesso, por exemplo, *firewalls* nas fronteiras da rede;
   - o aumento do monitoramento para detectar ou prevenir ataques reais (IDS ou IPS);
   - o aumento da conscientização sobre a vulnerabilidade;

h) seja mantido um registro de auditoria de todos os procedimentos realizados;

i) com a finalidade de assegurar a eficácia e a eficiência, deve ser monitorado e avaliado regularmente o processo de gestão de vulnerabilidades técnicas; e

j) deve-se abordar em primeiro lugar os sistemas com altos riscos.

4.6 CONTROLE E MATURIDADE DO PROCESSO

Os mecanismos de controle do processo de desenvolvimento seguro são realizados através da medição do nível de maturidade e acompanhamento por indicadores.

4.6.1 MEDIÇÃO DO NÍVEL DE MATURIDADE ATUAL DO PROCESSO

4.6.1.1 A maturidade deste processo é medida através da seguinte escala:

0 – Não Existente: A avaliação de requisitos e decisões sobre a utilização de sistemas seguros para a organização não ocorre. A organização não considera os impactos na organização associados a vulnerabilidades em sistemas e às incertezas de projetos de desenvolvimento de sistemas. O processo de desenvolvimento seguro não é identificado como relevante para a aquisição de soluções e para entregar os serviços de Tecnologia da Informação.

1 – Inicial/*Ad Hoc*: As vulnerabilidades e riscos dos sistemas de informação são levados em consideração de maneira *ad hoc*. As vulnerabilidades técnicas relacionadas aos sistemas de informação, como segurança, disponibilidade e integridade, são eventualmente levadas em consideração. Existe um entendimento emergente de que os requisitos de sistemas seguros são importantes e precisam ser levados em consideração.

2 – Repetível, mas Intuitivo: Uma abordagem de avaliação sobre requisitos de sistemas seguros em sistemas de informação imaturos e em desenvolvimento existe e está implementada. O gerenciamento de requisitos de sistemas seguros é normalmente em nível macro e está tipicamente aplicado apenas a projetos importantes ou em resposta a problemas.

Os processos de desenvolvimento seguro para sistemas de informação estão começando a ser implementados.

3 – Processo Definido: O gerenciamento de requisitos de sistemas seguros segue um processo definido e documentado. O treinamento em gerenciamento de requisitos de sistemas seguros está disponível para todos os usuários e desenvolvedores de sistemas. As decisões para seguir o processo de gerenciamento de requisitos de sistemas seguros e para receber treinamento são deixadas a critério individual. A metodologia para a avaliação de vulnerabilidades é convincente e bem estruturada e garante que os riscos-chave para os sistemas sejam identificados. Um processo para desenvolvimento seguro é normalmente instituído.

4 – Gerenciado e Mensurável: A avaliação e o gerenciamento de requisitos de sistemas seguros são procedimentos padrões. As exceções ao processo de gerenciamento de requisitos de sistemas seguros são relatadas. As vulnerabilidades são avaliadas em nível de projeto individual e também regularmente a respeito da operação de Tecnologia da Informação como um todo. Existe a capacidade de monitorar a posição dos riscos associados aos sistemas de informação e tomar decisões informadas referentes à exposição que se deseja assumir. Todas as vulnerabilidades identificadas têm um proprietário nomeado. Um banco de dados de sistemas da informação está estabelecido e parte dos processos de gerenciamento de requisitos de sistemas seguros começa a ser automatizado.

5 – Otimizado: O gerenciamento de requisitos de sistemas seguros já se desenvolveu a um estágio onde um processo estruturado é executado e bem gerenciado. Boas práticas são aplicadas através de toda a organização. A captura, a análise e o relatório de dados de gerenciamento de requisitos de sistemas seguros são altamente automatizados.

4.6.1.2 A tabela abaixo apresenta as metas para a evolução dos níveis de maturidade:

Nível de Maturidade	Metas	Prazo
Repetível, mas Intuitivo	• Possuir uma normativa interna do DECEA para desenvolvimento seguro. • Iniciar a implantação e testes do processo de desenvolvimento seguro em pelo menos 50% das Organizações Subordinadas ao DECEA.	Até junho de 2014
Processo Definido	• Implantar o processo de desenvolvimento seguro em todas as Organizações Subordinadas ao DECEA. • Capacitar todos os chefes das seções de segurança da informação.	Até junho de 2015
Gerenciado e Mensurável	• Criar um painel para acompanhamento, através de indicadores gerenciais do processo, a fim de garantir a tomada de decisão pela Direção do DECEA.	Até dezembro de 2015
Otimizado	• Realizar uma reunião semestral de análise crítica para melhoria contínua do processo de desenvolvimento seguro. • Possuir sistema informatizado para emissão de relatórios automatizados.	Até dezembro de 2016

4.6.2 ACOMPANHAMENTO DO PROCESSO POR INDICADORES

Objetivos do Processo	Indicadores do Processo
• Reduzir a ocorrência e impacto de vulnerabilidades técnicas encontradas em sistemas da informação; e • Determinar planos de ação com custos eficientes para vulnerabilidades e riscos críticos identificados em sistemas da informação.	• Quantidade de novas vulnerabilidades identificadas em sistemas da informação (comparado com o exercício anterior); • Quantidade de riscos identificados em sistemas da informação (comparado com o exercício anterior); • Quantidade de vulnerabilidades identificadas em sistemas da informação por nível de criticidade; • Percentual de vulnerabilidades críticas identificadas em sistemas da informação que possuem planos de ação desenvolvidos; • Quantidade de riscos identificados em sistemas da informação por nível de criticidade; e • Percentual de riscos críticos identificados em sistemas da informação que possuem planos de ação desenvolvidos.

4.7 FATORES CRÍTICOS DE SUCESSO

São os seguintes os fatores críticos de sucesso que deverão possibilitar o alcance dos objetivos definidos para o processo de desenvolvimento seguro, bem como nortear as avaliações dos resultados alcançados:

a) Garantia de cumprimento das responsabilidades atribuídas no processo;

b) garantia de cumprimento dos procedimentos relacionados ao processo;

c) acompanhamento da situação do processo e apresentação de relatórios periódicos;

d) garantia de comunicação eficiente e eficaz do processo para todas as partes interessadas e envolvidas; e

e) garantia de constante atualização quanto ao surgimento de novas vulnerabilidades e técnicas e riscos em sistemas da informação associados a sua exploração.

5 DISPOSIÇÕES FINAIS

5.1 O procedimento para aquisição e desenvolvimento de *software* seguro apresentado neste documento é de caráter geral e deve ser revisado periodicamente a cada vinte e quatro meses.

5.2 Esta Instrução deverá estar em conformidadecom as Diretrizes da DTI – Órgão Central do Sistema de Tecnologia da Aeronáutica –, e será revisada e atualizada sempre que forem atualizadas ou aprovadas Normas relativas ao assunto pela Diretoria de Tecnologia da Informação do Comando da Aeronáutica.

5.3 Casos não previstos nesta Instrução deverão ser levados à apreciação do Exmo. Sr. Diretor-Geral do DECEA.

REFERÊNCIAS

BRASIL. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS – ABNT NBR ISO/IEC. Sistema de Gestão de Segurança da Informação. Rio de Janeiro, RJ, 2005.

BRASIL. Clavis Segurança da Informação. Desenvolvimento Seguro – Security Development Lifecycle (SDL) – Conviso Security Training 201. Rio de Janeiro, RJ, 2013.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Plano Diretor de Segurança da Informação do DECEA*: **PCA 7-11**. Rio de Janeiro, RJ, 2010.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Política de Segurança da Informação do DECEA*: **DCA 7-2**. Rio de Janeiro, RJ, 2010.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Preceitos de Segurança da Informação do DECEA*: **ICA 7-19**. Rio de Janeiro, RJ, 2012.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Processo de Gestão de Incidentes de Segurança da Informação do DECEA*: **ICA 7-23**. Rio de Janeiro, RJ, 2013.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Processo de Gestão de Mudanças de Ativos de Tecnologia da Informação do DECEA*: **ICA 7-24**. Rio de Janeiro, RJ, 2013.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Processo de Gestão de Riscos de Segurança e Tecnologia da Informação do DECEA*: **ICA 7-26**. Rio de Janeiro, RJ, 2013.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Processo de Gestão de Vulnerabilidades de Ativos de Tecnologia de Informação do DECEA*: **ICA 7-27**. Rio de Janeiro, RJ, 2013.

BRASIL. Comando da Aeronáutica. Departamento de Controle do Espaço Aéreo. *Segurança da Informação e Defesa Cibernética nas Organizações do Comando da Aeronáutica*: **NSCA 7-13**. Rio de Janeiro, RJ, 2013.

BRASIL. Escola Superior de Redes. UTO, Nelson. Teste de Invasão de Aplicações Web. Rio de Janeiro, RJ, 2013.

BRASIL. Presidência da República. Gabinete de Segurança Institucional. Departamento de Segurança da Informação e Comunicações. Norma Complementar 16/IN01/DSIC/GSIPR – Diretrizes para Desenvolvimento e Obtenção de Software Seguro nos Órgãos e Entidades da Administração Pública Federal. Brasília, DF, 2012.

ESTADOS UNIDOS. CRC Press. MERKOW, Mark S.; RAGHAVAN, Lakshmikanth. Secure and Resilient Software, Requirements, Test Cases and Testing Methods. EUA, 2011.

ESTADOS UNIDOS. ISO/IEC 15408-1. Information Technology – Security Techniques – Evaluation Criteria for IT Security. EUA, 2009.

ESTADOS UNIDOS. ISO/IEC 27025. Information technology – Security techniques – Information security risk management. EUA, 2011.

ESTADOS UNIDOS. Microsoft Corporation. Implementação Simplificada do Microsoft SDL. EUA, 2010.

ESTADOS UNIDOS. Microsoft Press. HOWARD, Michael; LEBLANC, David. Writing Secure Code. 2. ed. EUA, 2003.

ESTADOS UNIDOS. OPEN Process Framework Repository Organization (OPFRO). Disponível em: <http://www.opfro.org>.